THE UNIVERSITY *of* EDINBURGH

This thesis has been submitted in fulfilment of the requirements for a postgraduate degree (e.g. PhD, MPhil, DClinPsychol) at the University of Edinburgh. Please note the following terms and conditions of use:

- This work is protected by copyright and other intellectual property rights, which are retained by the thesis author, unless otherwise stated.
- A copy can be downloaded for personal non-commercial research or study, without prior permission or charge.
- This thesis cannot be reproduced or quoted extensively from without first obtaining permission in writing from the author.
- The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the author.
- When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given.

A Methodology for the Production and Delivery of Generative Music for the Personal Listener: Systems for Realtime Generative Music Production

Michael J. Murphy

Doctor of Philosophy

The University of Edinburgh

2013

Abstract

This thesis will describe a system for the production of generative music through specific methodology, and provide an approach for the delivery of this material. The system and body of work will be targeted specifically at the personal listening audience. As the largest current consumer of music in all genres of music, this represents the largest and most applicable market to develop such a system for. By considering how recorded media compares to concert performance, it is possible to ascertain which attributes of performance may be translated to a generative media. In addition, an outline of how fixed media has changed how people listen to music directly will be considered. By looking at these concepts an attempt is made to create a system which satisfies societies need for music which is not only commodified and easily approached, but also closes the qualitative gap between a static delivery medium and concert based output. This is approached within the context of contemporary classical music. Furthermore, by considering the development and fragmentation of the personal listening audience through technological developments, a methodology for the delivery of generative media to a range of devices will be investigated. A body of musical work will be created which attempts to realise these goals in a qualitative fashion. These works will span the development of the composition methodology, and the algorithmic methods covered. A conclusion based on the possibilities of each system with regard to its qualitative output will form the basis for evaluation. As this investigation is seated within the field of music, the musical output and composition methodology will be considered as the primary deciding factor of a system's feasibility. The contribution of this research to the field will be a methodology for the composition and production of algorithmic music in realtime, and a feasible method for the delivery of this music to a wide audience.

Lay Summary

This thesis will describe a method for the production of algorithmic music in realtime. The music produced will utilise stochastic algorithms to emphasise attributes of performance usually constrained to live performance. The focus is upon shifting trends in music consumption, primarily regarding the growth of the personal listening audience. In this context personal listeners are considered to be those who consume music at home, in their car, or on a mobile device. The specific group of listeners targeted by this research are those who consume contemporary classical music. By considering how this music is reproduced using recorded media such as compact disk, tape, or vinyl, and how this compares with concert performance, it is possible to ascertain some areas in which to apply algorithmic methods to music production. These algorithmic methods will attempt to close the gap between the reproduction of a CD or tape, and that of a concert performance. A portfolio of musical works will be created which will demonstrate this in practical form, and a way of delivering this music to the audience will be considered. These musical works will span the development of the research, and they will be used as one of the main elements for software evaluation. What is different about this thesis is a concentration upon musical works created through generative means which fit into a more "traditional" temporal archetype. This means works which fit into a standard format. For example, pieces with a beginning, middle, and end, rather than any esoteric or extended "infinite" length formats. In essence, a similar core musical material to that which would be delivered using a static medium, just using software to achieve this. Software is used as it is able to retain the contingent qualities of music that generative music allows. This research looks to the future of music delivery and composition methodology, while retaining specific traditional musical constructs.

Acknowledgments

Many thanks to my supervisors Dr. Michael Edwards and Prof. Peter Nelson for their support throughout this work. Also to Renee Timmers for allowing me to use the Disklavier at Sheffield University to record my work.

For their fantastic hospitality I have to thank the staff of the Experimentalstudio des SWR, the time I spent at the studio imparted a large chunk of inspiration towards the following projects. In particular Thomas Hummel, whose software conTimbre is now a major element of my ongoing work.

Contents

1 Introduction

The work presented in this thesis is an examination of a number of composition systems with respect to their accompanying musical work. These systems represent a chronology of work trending towards a standalone system for the creation and reproduction of generative concert music. This musical work may be written for an automatic performer such as an automatic piano, sampler, synthesiser, robot, or live-electronics. The goal is not to claim that an automatic performer may replace a human one, but to investigate the myriad of possibilities available with this technique of reproduction. To cover the field fully, an investigation into interactive, real time, and non-realtime applications was made. This has resulted in a number of software applications which accompany this thesis. The structure and application of these systems will be looked at in the relevant chapters.

The context for the compositions and software systems is that of music consumption in contemporary society. In particular, the way in which a large proportion of the overall audience consumes music will be considered. Also, a review of the way algorithmic and generative music can change the way an audience perceives music will be looked at. This will investigate how the audience for music has moved from live performance,[1] to a static recorded medium such as a compact disk. An exploration of selected software-based methods to restore the experiential attributes of live performance for recorded media are looked at. These cover constraint and pattern based methods of composition coupled with alternative virtual mediums for the release of contemporary music in society.

It has been said that new musical applications are rated primarily upon their technical achievements, instead of their musical application (Stroppa, 1999). To avoid this, systems in this report are evaluated primarily with respect to their specific musical application and ease of use. Due to the nature of the field, technical criteria ultimately play a large role in the musical output so these will be considered with this in mind. In addition to the software systems and their output, there are standard scores for my submitted piano pieces.

[1] That may be in a concert or the home environment

Fundamentally, the technical work in this thesis is based largely around deterministic and non-deterministic methods of composition through software systems. The investigation is into how a composer can develop systems using specific methodology to satisfy criteria for different means. There are examples of systems which cover many applications such as interactive, real time, and non-realtime systems. Primarily the goal is to achieve a standalone system for the reproduction and delivery of generative music, which should retain attributes usually reserved to a live performance. This is framed within the context of an ongoing paradigm shift in the generation and release of musical media.

1.1 Personal Motivation

To provide further context for the work contained in this thesis, it seems appropriate to provide my background and motivations for undertaking this project. This section may provide some insight into some of the decisions made with regard to the route in which the research took.

My personal background is that of an instrumental performer, and software developer. Computers have been present in my learning since an early age, and since learning BASIC in the mid 90s I have been fascinated with programming. This interest in software development later moved on to C++, with which I created many small programs. In 2005, after studying for my UK A-Levels I decided to change the direction my education was travelling and enroll for a course in jazz and world music for my degree. While studying for this I was creating electronic music in my spare time, and playing bass in various bands. Invariably I attempted to apply my knowledge of jazz harmony to the electronic music I was composing, but the improvisatory element was missing, along with the experiential quality of live performance. While considering the subject for my dissertation, the composer Iannis Xenakis was suggested to me as a possible topic. After completing the research for the dissertation, it became clear that algorithmic composition was the perfect mix of mathematics, software development, and composition.

From this starting block I continued to study the works of Xenakis and his compositional techniques. This lead me to the language MaxMSP, in which I could implement some of the techniques I had learned and investigate new systems of my own. Further investigation and advice directed me towards the languages Common Music and SuperCollider. Using these the main body of work for the PhD was completed. However, certain decisions about the level of algorithmic control were retained from my studies of Xenakis, who routinely modified the output of his algorithms to fit a compositional framework.[2] This concept is evident throughout the composition portfolio included. All of the pieces retain some discretionary elements, most clearly this is seen in the piano pieces which were largely edited by hand. However, even the final pieces such as *Stratus* have specific elements which are hard-coded.

1.2 Questions, Goals, Evaluation, Implementation

This thesis covers a fairly large area of study. Therefore, to determine the research questions and provide a clear path for investigation, the context must first be considered. Primarily, this research is centred around the personal listener. This could be either an individual, or a small group of people. Contrary to the traditional large concert hall audience, this research concentrates upon a decentralised network of listeners. More specifically, listeners of contemporary music in this environment. Traditionally, classical music music is listened to in a concert hall, or at home through a static medium such as compact disk or vinyl. However, recent changes in technology have lead to digital mediums increasing in popularity. In addition to this the internet is now available to many people wherever they have access to a wireless network, allowing access to streaming digital media. Therefore, the main question we approach is **how can the current personal listening medium be improved?** If concert hall performance is considered to be the standard in contemporary music delivery, this can be used as a way to determine what can be improved the personal listening context. The

[2] One example of this is the modification of Cellular automaton derived orchestration in Xenakis' *Horos*. Xenakis' modification of *Horos* was due to the periodicity of the Cellular automaton used, his editing was in order to remove the repetition this would lead to (Hoffmann, 2002)

main difference approached throughout this thesis is that of *performance*.[3] To achieve this concept of musical "performance" for the personal listener, generative music techniques are utilised.

1.2.1 Research Goals

With the context of improving the personal listening medium now in place, the goals necessary in order to achieve this can now be considered. These are outlined below, and reconsidered in the final chapter 6.

- **Create** a system for the reproduction of generative "concert" music for the personal listener
- **Develop** the system, and incorporate a feasible delivery method
- **Contextualise research within society** and provide realistic goals for the delivery of the music and algorithms created to an audience
- **Utilise** the systems created to generate a body of creative musical work
- **Contextualise research within the field** and develop creative work of a formalised nature within a predefined genre: contemporary classical music

These goals are based largely within the context of media production and delivery. Fundamentally, the aim is to produce musical works which retain consonant compositional constructs while being completely dynamic in nature. This is based around an alternate method for the reproduction of media. Essentially instead of concentrating on a static medium such as a compact disk, a software approach is taken. This is achieved by utilising standalone composition systems in the software environments discussed in further chapters. In fact this direction is already being taken in the mobile systems arena, wherein the mp3 is being superseded by the 'app'. In this environment musical scores, systems for

[3] This is covered in much more detail in section 2. In this case, and occasionally throughout the research, the term "performance" is used to describe the specific contingent *attributes* of a performance that give it an individual quality. By the term "performance" in this context, these attributes are what is considered.

composition and instruments are integrated with music so tightly they could be considered to be musical pieces themselves (Thor Magnusson, 2011). A question can be raised as to whether this approach enhances the *experiential* quality of music within the context of musical performance. In contrast to the static forms delivered through conventional media, this method would give an individuality to each reproduction. The goal would be to create a system which retains some of the contingent attributes of a standard performance, which are commonplace within classical music approached from the viewpoint of concert based output. Clearly when Rachmaninov plays Rachmaninov it will be noticeably different to when Ashkenazy plays Rachmaninov. These performances will not only different for each performer, but on each performance by the two performers. This is the sort of variation which provides the core of the investigation. To attempt to reintroduce this variation to the home listener and provide a dynamic software approach to this concept.

Some technical considerations must be made when thinking of a system such as this. The primary concern is that if there are no human performers, a decision must be made on how the sound will be generated. Here a number of possible options exist which have been achieved previously in various circumstances. The first possible option is to generate the work based upon a large body instrumental samples. One may attempt to incorporate new physical modelling techniques to generate orchestral instrument sounds. Conversely one may generate purely synthesised tones, or manipulate samples, and release music based around a tape paradigm. Another, more radical solution is to incorporate robotic instruments to play the pieces. This could take the form of automatic pianos or a large robotic orchestra such as that found at the Logos foundation.[4] One possible advantage of a robotic orchestra style output is that it may be applicable to concert based application. Indeed, some advantages are available to the composer of music for automatic instruments which will be discussed in the relevant chapters. However, a question can be raised over the aesthetic quality of a robotic performer over a human counterpart in a concert situation. Would an algorithmic piece benefit

[4] `http://www.logosfoundation.org/`: accessed 19/02/11

from a truly individual *generative* performance, over a standard static performance with a human performer? Here another question can be raised: what other methods for creation of a purely generative performance are there, and to what extent are aesthetic and compositional constraints imposed through the considered methodology? This is discussed further throughout the thesis.

1.2.2 Methodology

Throughout the research period, the method of working was based largely upon reflection and introspection upon projects completed. For example, a system may be created which facilitates the generation of a piece of music such as *pp*. However, throughout the composition of *pp* a number of bugs or areas for improvement are noted. By appraising the completed work with respect to the musical output and the ease of compositional use, it is possible to ascertain areas in which the system could be improved. In the case of *pp*, these areas were the panning and audio routing systems. In *For Putten* these areas were fixed and the software evolved according to the experience gained through the composition of *pp*. This can be thought of as an interactive cycle where conclusions and further questions emerge as the work develops. A similar methodology prevails over all of the research material, including standalone classes and software systems such as the *Stochastic Sampler*. A simple flowchart of the methodology can be described:

1. **Investigation** into a topic such as adaptive granular synthesis or compositional technique.

2. **Experimentation** may come in the form of a piece of software or a composition.

3. **Introspection** look at which elements worked and which require further thinking.

4. **Conclusion** how these issues can be rectified in the following version.

1.2.3 Evaluation

The work accompanying this thesis spans many different procedures for composition. Due to this, the evaluation criteria must adapt to the development process which is outlined in each broad section of the thesis. This can be broken down into two major elements: realtime, and non-realtime. Much of the work created for this thesis was working towards a final goal: a software methodology for the production and delivery of generative music. This has lead to a bias towards realtime constructs. In the case of these realtime constructs, the software design forms a large part of the evaluation procedure. This is because in a realtime system there is no time for discretionary editing, the software must perform as intended every time without fail. Conversely, in a non-realtime system such as that created for *SnSu*, the software output was a base for editing rather than the musical goal.

The most important element of each of the systems is their individual aesthetic output, and conciseness of workflow. Fundamentally, regardless of compositional technique the goal is to produce relatively aesthetically pleasing output. Evaluation of the aesthetic merit of a piece of music is comparatively more difficult than determining the functionality of a software system. Therefore this is approached by first looking at the strategy used to determine the output, usually an algorithmic construct represented through code, and evaluating the output in musical terms. This places many of the pieces within section 5.4 which focuses on *patterns*, as this was the fundamental construct used to generate much of the material.

In addition to the technical and aesthetic aspects, the experiential advantages (or disadvantages) of a dynamic technique over conventional methods must be evaluated in the context of media consumption. As the larger market share is inarguably stacked toward static rather than generative media, the advantages of the proposed method of reproduction can be argued. However with the rising trend of the 'app' over many conventionally static forms of media delivery, a possible paradigm shift approaches. For this investigation, algorithmically designed classical music provides the platform for an application of this concept.

Criteria for the evaluation of completed work was dependent upon a number of elements. For a system creating a piece of music it may be the usability and ease of composition. The piece *pp* can be looked at as an example. With this piece, panning was a difficult process due to the way in which is was implemented. This raised usability issues that were therefore fixed in the next version. Development of new software may often raise more questions than it answers; examples that occurred throughout this research period were usually related to **efficiency**, **usability**, and **audio output**. These criteria would usually be thought of during the "introspection" phase of the methodology described in the previous section. Another example to show how these criteria affected the development of a piece of software is the *Stochastic Sampler*. Here a piece of software was created within MaxMSP which formed an investigation into adaptive granular synthesis by means of Markov chain. This software needed further work on two fronts, usability, and efficiency. Therefore a Java implementation was completed which solved these two problems yet raised another, audio quality. The Java implementation had bugs which lead to issues with audio output. In addition to this it did not fit with the composition framework which was now within the SuperCollider environment, which raises issues with usability. Therefore a third iteration was completed which solves all of the aforementioned issues. However, In addition to the criteria of efficiency, usability, and audio output, some more general criteria can be considered when looking at a project such as this. For example, **feasibility**. This came up in a number of contexts, such as when considering about methods for the reproduction of generative music. A number of different concepts were considered, but after research into different methodologies for achieving this, a sampling method was settled upon. This was largely down to problems with the feasibility of the other options, not necessarily any inherent problems with the approach itself. Another more general criteria, more focused upon the software development side of the project would be **scalability**. Some of the software developed throughout the research started off relatively large and difficult to expand upon. By creating systems that can be developed and expanded over time, a good

Table 1: Software created throughout the research.

Software	Environment	Medium
Stochastic Sampler (2009-10)	MaxMSP, Java	Installation
Stochastic Sampler 2 (2012)	SuperCollider	Installation / DSP
CA Sampler (2009-10)	MaxMSP, Java	Installation
CA Sampler 2 (2012)	SuperCollider	Installation / DSP
Composition		
Warblers (2009)	MaxMSP	Automatic Piano
SnSu (2010-11)	Common Music	Automatic Piano
Prime Pattern 33 (2009)	Common Music	Automatic Piano
Hermit (2011)	Common Music	Automatic Piano
pp (2011)	SuperCollider	Automatic Piano / Sampler
Wet (2010)	Ableton Live	Tape
Traurig? (2010)	SuperCollider	Tape
For Putten (2012)	SuperCollider	Sampler
Stratus (2012)	SuperCollider	Sampler

base for continued research from a well established platform can be built. This method of reflection and development upon the criteria of usability, efficiency, and audio output, forms the basis for the progress of the research. In addition to this it provides a base for evaluation in the conclusion.

1.2.4 Software Implementation

In this section, how the concepts discussed throughout the thesis were implemented will be considered. Throughout the investigation, a number of different software environments were utilised. Currently the most dominant of the platforms for computer music are developed for DJ mixing or sequencing. Tools such as Ableton Live, Cubase, or Traktor fit into this archetype. Other softwares that allow for the creation of synthesisers and interfaces such as MaxMSP and Reaktor also exist, and provide for the creation of algorithmically composed work. However arguably the most powerful of the environments are full programming languages with audio libraries such as Csound, SuperCollider, or Common Lisp Music (CLM). All three families of software were utilised in the creation of the work accompanying this thesis.

Much of the current use of computers in live performance utilises the computer as a powerful effects machine. This research aims to investigate different ways of creating music using the computer, using it not only as a signal processing system but also as a platform for generative composition. The bulk of the compositional work done for this report used the MaxMSP, Common Music, and SuperCollider environments. These three environments have specific advantages for the different styles of workflow covered in this thesis, and will now be discussed.

MaxMSP was used in this project to develop systems for granular synthesis along with numerous experiments in algorithmic composition. The creator of Max, Miller Puckette, describes the Max system as "a way of combining pre-designed building blocks into configurations useful for real-time computer music performance" (Puckette, 2002). One of the advantages of the Max environment is ease of use and simple interface with Java and C++. This is important, as efficiency can be an issue when designing computer music systems. Due to this, an advantage of using the Max environment is the quick and simple integration of the Java Virtual Machine through the `mxj` and `mxj~` objects. Using these, the efficiency of complex systems was increased substantially.[5] If C++ is considered, an even larger leap in efficiency is available. Max/Java were used to create the first and second versions of the automatic sampling system, along with the Markov chain and cellular automata implementation. The *jMegaHal*[6] API was used within Java to create the Markov implementation, in addition to this the external pitch-tracking object `fiddle~`[7] was used to generate the basis for the piece *Warblers*. These are discussed in depth in the relevant sections.

SuperCollider was used to generate many of the pieces, and the main body of work. James McCartney describes SuperCollider as "a dynamically typed, single-inheritance, single-argument dispatch, garbage-collected, object-oriented language similar to Smalltalk" (McCartney, 2002; James McCartney, 1998). SuperCollider allows for complex nested pattern-based methods of composition along with signal processing capabilities. SuperCollider has some specific advantages for my personal workflow. One of these is that efficiency in SuperCollider

[5] See section 3.4 for details.
[6] `http://www.jibble.org/jmegahal/` accessed: 02/02/12
[7] `http://crca.ucsd.edu/~tapel/software.html/` accessed: 02/02/12

can be easily achieved, signal processing elements are sent to the audio server when they are required. Due to the simple way one can control synthesiser definitions and the ability to experiment quickly, much of the more complex signal processing work was completed in this environment. SuperCollider was a perfect tool to design the synthesis based live electronic pieces because of the synthesis backend and pattern-based composition tools available. This allowed for patterns driving the instruments to be synchronised perfectly in time with signal processing elements, without the need for any external equipment.

Common Music was used to create the basis for the pieces *SnSu*, *Prime Pattern 33*, and *Hermit*. Common Music is a system which produces sound "by transforming a metalevel representation of music into a variety of different protocols for controlling sound production and display" (Heinrich K. Taube, 2004; Heinrich Taube, 1991). The languages used to write music within the Common Music environment are Common Lisp, Scheme, or Sal2. Lisp and the languages that have followed from it are well known to be perfect for dealing with lists, in fact the name 'Lisp' was derived from the term 'LISt Processing' (Mitchell Wand, 1984). Dealing with arrays and lists of elements is fundamental for a language working with composition and computer music. Due to the large amount of array data manipulated to create the piano pieces, Common Music was the ideal choice to realise these compositions. In addition to this, the multitude of music-oriented functions contained in the Common Music package aid experimentation immensely. One example of this is the quick way a composer can generate material for review in this environment. It is an extremely quick process to generate multiple 'takes' in MIDI format, these files may then may be reviewed and edited.

In addition to these pieces of software, much non-realtime and post-processing work was completed using the digital audio workstation softwares Reaper and Ableton Live. The scores for the piano pieces were generated in Sibelius after heavy discretionary editing and modification. To generate acoustic recordings of the piano pieces, a Yamaha Disklavier was used with the kind permission of Sheffield University. The prepared piano pieces were generated using the Native Instruments Kontakt sampler along with the John Cage prepared piano library.

For *Stratus* and *For Putten* the orchestration software conTimbre[8] was utilised.

Accompanying this thesis is a full disclosure of all SuperCollider, Common Music, MaxMSP patches, and Java code. As only the main concepts will be covered in this thesis, if the reader wishes to investigate the systems in further detail they are included in the appendices and accompanying material.

1.3 Thesis Structure

Each chapter within this thesis focuses on a specific aspect of computer music composition. The application of these techniques to the compositions in the portfolio is discussed in the relevant sections. The sections cover many different aspects of computer music composition and delivery. A further section investigates perception of music, looking at how people consume the music they listen to from a Western perspective. This examines how a generative piece may affect the quality of the listening experience for a respective audience.

As many of the systems developed were constrained to producing a single composition, one of the core goals was to investigate how certain methodology could lead to interesting compositional practice. This was approached in a non-realtime context therefore is found in section 5. Ultimately, a system for the production and distribution of generative music was completed. This became the focus of the study on the more technical aspects of generative composition methodology. In keeping with the fundamental goal of qualitative output, development of this system is evaluated with respect to the musical output. The system and its technical implementation is discussed in depth in section 4.1.

Signal processing was approached throughout as an extension of the synthesised or acoustic sound, rather than a separate element. Throughout section 4, the techniques used to create the signal processing elements and technical aspects will be discussed. Here the compositions will be approached from a more technical point of view, by dissecting the signal processing path to show how these techniques were utilised in composition. As two different signal processing environments were utilised throughout the course of research, a similar method

[8] `http://www.contimbre.com/` accessed: 13/11/12

of presentation will be applied where possible.

An example of a large MaxMSP implementation is the research completed into interactive systems for music. This research is covered in section 3. Here the systems were developed using a combination of MaxMSP and Java. An investigation was made into a specific methodology for stochastic granular synthesis. This was developed further by introducing finite state systems and cellular automata as control methods for the synthesis system. Primarily the goal was to create a system for the reproduction of generative music which would double as a signal processing tool where necessary.

Perhaps most importantly, the context of the work completed in this thesis is covered in section 2. Here the way a listener may perceive and consume music is investigated. This is looked at with regard to the production of generative music, and what advantages or disadvantages it may have over a traditional format. Finally, the conclusion aims to bring all of these interrelated threads together to a concise result. In this section the work completed and how further developments could be made will be considered. In addition, some evaluation of the systems from a technical standing will be considered. This will consider how these systems could be applied in a different way to achieve varying qualitative output.

1.4 Background

To provide some background to the work completed for this thesis, an overview of the history of the approach will now be considered. Algorithmic music has been prevalent through musical history. It can be described as the formalisation of construction processes with which music can be created (Hedelin, 2008). It could also be described as a "step-by-step" recipe for creating new compositions (Muscutt and Cope, 2007). It does not need to be automated, and does not have to be created with a computer. However, multiple possibilities may be explored quickly through automation, and avenues of complex compositional techniques can be simplified through software abstraction. Machines can allow for complex signal processing and synthesis techniques to generate new timbres, along with performance capabilities that transcend that which a human performer could achieve.

1.4.1 Algorithmic Composition

Mathematical constructs have been present in music since Pythagoras, who determined the harmonic ratios and thus early music theory (Richard L. Crocker, 1963, 1964). Ancient music has strong links with systematic compositional procedure. This can be seen through Gregorian chant, which was founded upon the structure of ancient music. The modes of Gregorian chant can be characterised as melodic formulae rather than proto-scales. This can be taken further to affirm that "ancient music, at least up to the first centuries of Christianity, was not based at all on scales and 'modes' related to the octave, but on tetrachords and 'systems'" (Xenakis, 1970, pg. 4). These hierarchic structures can be explained through examining four *orders* apparent in the music of ancient Greece. After Aristoxenos all ancient texts show this formalised hierarchical procedure, covered in Xenakis (1970) as the following:

- The *primary order*; the tone and its subdivisions. This is defined as the amount by which the interval of a fifth exceeds the interval of a fourth. The tone is divided into halves, thirds, and quarters. These are known as *semitones*, *chromatic dieseis*, and *enharmonic dieseis*.

- The *secondary order*; the tetrachord. The tetrachord is bounded by an interval of two and a half tones: a *dia tessaron*, or fourth. Outer notes maintain the same interval while the inner notes are mobile.

- The *tertiary order*; a combination of elements of the first two tones and tetrachords. These are either conjunctival or separated by a tone. This creates a pentachord, wherein the outer interval is a perfect fifth, and the octochord, where the outer interval is an octave. Subdivisions of the system follow that of the tetrachord.

- The *quaternary order*; the tropes, keys, modes. These are derived from cadential, melodic, dominant, registral, and other formulae.

This Hellenic hierarchical tree was completed by a set of *transition algorithms*, from each system to another or from one mode to another (Xenakis, 1970). With regard to notation, formalisation stems back to at least 1026 and the composer Guido d'Arezzo, the inventor of staff notation. D'Arezzo developed a formal technique to set text to music (Michael Edwards, 2011; Jos. Smits van Waesberghe, 1951). This scheme, defined in his *Micrologus*, assigned a pitch to each vowel so that a melody was created based upon the vowel content of the text (Christopher Ariza, 2011).

Further algorithmic approaches can be found in early music technique. A term known as "isorhythm" was invented at the beginning of this century by Friedrich Ludwig[9] (Denis Harbinson, 1966). One early example of this is apparent in the masses, motets, and other sacred music of the 14$^{\text{th}}$-century composer Guillaume de Machaut (Alice V. Clark, 2004). The isorhythmic technique is based upon a system wherein rhythmic cycles, known as *talea*, combine with melodic cycles, called *colour*. These cycles can be of the same or differing length. This difference in length can potentially lead to long modulated forms, moving in and out of sync with each other over time.

Following from this, another example is the composer Guillaume Dufay (1400-1474), who derived parameters for his compositions based upon number ratios. The tempos Dufay used for one of his motets were derived from the proportions of a Florentine cathedral. His piece, *Nuper Rosarum Flores* (1436) attempted to reference the cathedral's essence through this method; it was Dufay's musical dedication to the Cathedral of S. Maria del Fiore (Marvin Trachtenberg, 2001). The temporal structure of this motet was based on the ratios 6 : 4 : 2 : 3, said to be the proportions of the nave, the crossing, the apse, and the height of the cathedral (Michael Edwards, 2011; Charles W. Warren, 1973). There have been some critiques of this analysis, most specifically by Craig Wright, who states "the unique ratio 6 : 4 : 2 : 3 which governs Dufay's motet, is, however, in no way immanent, or even superficially apparent, in the design of the cathedral of

[9] Guillaume de Machaut, Musikalische Werke, ed. Friedrich Ludwig, Publikationen älterer Musik 3 (Leipzig: Breitkopf & Härtel, 1929)

Florence" Craig Wright (1994) quoted in (Marvin Trachtenberg, 2001, pg. 744). Nevertheless, *Nuper Rosarum Flores* has a systematic, architectonic design, and provides an important historical landmark in the history of formalised music.

Dufay also applied highly systematic procedures such as inversion and retrograde to tone sequences (Curtis Roads, 1996). Furthermore, he has been found to have utilised the Fibonacci ratio or *golden section* in the composition of a number of his motets (Margaret Vardell Sandresky, 1981). Fibonacci relationships have been found in the music of Bach, Schubert, and Bartók, as well as a large number of 20$^{\text{th}}$ century composers (Michael Edwards, 2011).

Even a relatively well ingrained system such the fugue is an example of a fixed structure. It is an automatism, which was utilised two centuries before the birth of the theory of abstract automata. It can even be seen as the first automaton (Xenakis *et al.*, 1987). Therefore the fugues, canons, and similar forms of Bach can also be examined from an algorithmic standpoint (Muscutt and Cope, 2007). One particularly famous example of the early application of probability theory to music are the *Musicalisches Würfelspiel* (musical dice games) of Mozart and Haydn. The permutations of the dice in this example are self-contained compositional units which are designed to link coherently in all possible combinations (Manning, 1980). The element of chance incorporated into these dice games is something that has continued to be prevalent in algorithmic music to this day.

1.4.2 Automated Composition

Examples of automated composition can be found from as early as 1956. One such example is a system known as *DATATRON*, which was used by Martin Klein and Douglas Barrows to create 'Tin Pan Alley' melodies. One such result of this was the melody *Push Button Bertha* in 1956 (Ames, 1987; Christopher Ariza, 2011). Some of the most well known computer aided compositional experiments were also in process during this period. Hiller, Isaacson and Baker's *Illiac* computer was designed and built in Urbana, Illinois. Their *Illiac Suite* for string quartet was composed using two basic approaches: random selection constrained by lists of rules, and Markov chains, wherein the likelihood of an event is determined by a preceding number of events (Ames, 1987).

Another of Hiller's collaborators was Robert Baker, the creator of the composing utility *MUSICOMP* (MUsic Simulator Interpreter for COMpositional Procedures). *MUSICOMP* allowed for the process of developing new composing programs by managing libraries of compositional subroutines, which composers could link together in a main program designed to meet their compositional goals. Hiller and Baker's *Computer Cantana* was the result of a series of studies in computer music composition carried out in 1963. The studies were to test the efficiency and ease of use of *MUSICOMP*. The *Computer Cantana* was built from eleven separate sections, grouped into a five-movement performance plan (Hiller and Baker, 1964). The sections moved through systems of random selection, pre-determined instrumental music, and Markov chain based material with varied weightings. In addition to this, *Computer Cantana* employed serial methods drawn from Pierre Boulez's *Structures* for two pianos. This was in addition to the stochastic methods employed to create the *Illiac Suite* (Ames, 1987).

Continuing with the theme of stochastic methods for composition, Iannis Xenakis is one of the most well-known composers to utilise algorithmic and automated systems for composition. An architect, engineer, and composer, Xenakis was a pioneer of algorithmic and computer composition. His *Stochastic Music Program* was first published in his book (Xenakis, 1992), and utilised the Maxwell-Boltzmann distribution.[10] This program was used to compose *Pithoprakta*, which utilised these formulae to create clouds of sound with the orchestra that were to represent an analogue to the movement of particles in a gas (Xenakis, 1992). The *Stochastic Music Program* models a composition as a sequence of sections. Each section is differentiated by its duration and the density of events contained within. A composer works with the *Stochastic Music Program* by modifying certain global parameters, then executing the program. The global attributes that can be modified are: average duration of sections; minimum and maximum density of notes in a section; classification of instruments into tim-

[10] In thermodynamics, the distributions of energies of the particles in any gas are given by the Maxwell-Boltzmann distribution

bre classes; distribution of timbre classes as a function of density; probability for each instrument in a timbre class to play, and the longest duration playable by each instrument (Curtis Roads, 1996). Using these parameters, Xenakis was able to produce the basis for a composition. Later, the composer John Myhill would improve the *Stochastic Music Program* and re-code it for use with personal computers.

An expansion on Xenakis' *Stochastic Music Program* would come later, in the form of a system which allowed composition of both macro and micro elements stochastically. In 1991 at CEMAMu,[11] Xenakis wrote a program in BASIC that runs on a PC. The program was called *GENDY*: GEN stands for Generation, and DY for Dynamic (Serra, 1993; Xenakis, 1992). *GENDY3* was a stochastic work entirely produced by *GENDY*. In *GENDY3* the sound synthesis and musical structure are both based upon a stochastic algorithm which Xenakis invented and named "dynamic stochastic synthesis". To create a stochastic timbre, Xenakis would work completely within the time domain instead of resorting to spectral composition. First, the dynamic stochastic synthesis model would create a random sound. From there it computes each individual waveform by applying stochastic variations to the waveform preceding it. On each repetition of the system the frequency and amplitude of the given waveform is distorted by the stochastic algorithms, this creates a stochastic timbre (Serra, 1993; Peter Hoffmann, 2000).

Another important proponent and creator of automatic music composition systems is the composer Gottfried Michael Koenig. Koenig described his understanding of computer composition as "the formulation of sets of rules with the aid of a computer with a view to working out musical contexts without explicitly defining the acoustic presentation space" (Koenig, 1991). His work began in 1964 on *Project 1*, a Fortran program which "described a generalised model of serial composition" (Koenig, 1991, pg. 175). Koenig's background was in serial composition, working as a composer, assistant, and teacher at the Cologne electronic music studio. However by the time he had started working on *Project 1*,

[11] Centre d'Ètudes de Mathématique et Automatique Musicales (France)

Koenig realised that "the trouble taken by the composer with the series and their permutations has been in vain; in the end it is the statistical distribution that determines the composition" Gottfried Michael Koenig (1970) quoted in (Ames, 1987, pg. 176). *Project 1* was responsible for several of Koenig's composition including the 1965 *Project 1, Version 1* for 14 instruments.

Project 2 was first created in 1970. This differed from the earlier version as it was designed for general use, not only as a personal composing system. Indeed, Koenig specifically encouraged others to develop digital synthesisers which would accept data prepared via his program (Manning, 1980). *Project 2* incorporated a broad statistical palette of procedures which users could patch together in different combinations. Although it was developed primarily for pedagogic functions, it was been used in one instance to create the piece *Übung für Klavier* ("Study for Piano") in 1970. The score for this work consists of 12 'structures', each of which appears in three variants, for which the only change to *Project 2* is the duration. Koenig would never consider his *Project 1* and *Project 2* to be complete systems for composition. Instead, he acknowledged the creative insight a composer may find during the process of transcribing the numerical lists generated by the systems he created. In his 1991 article, Koenig states "the interpretation of the score table serves the purpose of revealing the musical idea on which the input data are based; not the idea for a particular piece, perhaps, but for composition itself" (Koenig, 1991, pg. 177).

Despite all of this, the perception that algorithmic composition lacks inspiration and personal involvement persists throughout the history of algorithmic composition (Muscutt and Cope, 2007). It is often looked upon as a sideline in contemporary musical activity, rather than an application of a compositional technique into the digital domain (Michael Edwards, 2011).

1.4.3 Automatic Performance

Until the middle of the 20th century, most automatic musical instruments were either mechanical or pneumatic (Ajay Kapur, 2005). The mechanics involved allowed for relatively precise timings, however dynamics, and timbral possibilities were extremely limited. Primarily, early musical robotics were focused on mechanical keyboard instruments. An example of this is the Pianista (1863) developed by French inventor Jean Louis Nestor Fourneaux. A modern day equivalent of this would be the Disklavier, with which the recordings of the piano pieces accompanying this thesis were rendered. The advent of electromechanics has allowed for far more versatility in automated musical instruments. Through new interest the field has moved through other instruments such as chordophones, aerophones, membranophones, and idiophones (Weinberg and Driscoll, 2006). There has also been research into anthropomorphic designs (Roads, 1986). A comprehensive overview of the history of musical robotics has been completed in Ajay Kapur (2005).

Contemporary work in the field of musical robotics fall into overlapping categories. The first of these are industrial anthropomorphic robots, designed to explore the way humans play musical instruments. An example of an anthropomorphically designed robot is the Tsukuba musical robot. It was designed to play the organ in the same way as a human performer would. The 90 kg robot, equipped with a video camera for eyes, could also analyse and perform from a piece of sheet music. In addition to this the robot is able to track a human singer, and play the organ along with the performer. The pitch is attained through a system of five narrowly tuned bandpass filters, sampling every 30 ms (Roads, 1986). Another example of an anthropomorphic robot is the Waseda Flutist Robot: WF-4RII. In 1990, research began into an anthropomorphic flute playing robot at Waseda University. The goal of the research was to understand further the motor control process required to play the flute. The first version was developed in 1990, and reproduced the human lung system by using a bellowphragm with a piston and cylinder mechanism (Jorge Solis *et al.*, 2006). The robot could synchronously perform with MIDI accompaniment data by combining its control system with a MIDI-processing unit. This was further developed to incorporate a system for embouchure control.

There are also instruments that are designed to serve as interactive agents to explore human-machine interaction. Mari Kimura's piece *GuitarBotana*, written in 2004 is a work for violin and robotic guitar. GuitarBot was designed by Eric Singer with LEMUR, and is based on the slide guitar. Each string of GuitarBot has an individual plucking device. The rotary picking mechanism is known as a "PickWheel", which is a series of three picks that rotate at a given speed. Fretting of the instrument is achieved through use of a movable bridge which travels along the length of the neck of the instrument, with a damper solenoid at one end (Ajay Kapur, 2005). Both the sliding and picking mechanism are controlled by DC servo motors and the entire system is controlled via the MIDI protocol (Weinberg and Driscoll, 2006). The piece is designed so that sections of improvisation between Kimura and Guitarbot and sections of score following are interlaced. This is achievable through the software Kimura has designed for the composition. In some cases GuitarBot follows Kimura's playing closely, and in others it is programmed to disregard pitches played by the violinist and produce more unpredictable output (Auslander, 2009). This is an example of algorithmic music designed for a robotic instrument, with interactive improvisatory qualities.

Another field of contemporary musical robotics include those that seek to explore the unique capabilities of the robotics themselves. One such example is the Logos Man and Machine Ensemble in Ghent, Belgium. The Logos foundation started in 1968 as a collective of composers and musicians. Originally concentrating on electronic sound generation devices, the construction of an automatic acoustic saxophone shifted the focus onto robotic musicianship (Laura Maes *et al.*, 2011). The orchestra contains many classes of instruments including organ instruments, string instruments, percussion instruments, and noise generators.

One of the most well-known composers to utilise musical automata is Conlon Nancarrow. His studies for player piano vary in style and form radically, even though 75% of his output is for a single instrument. Nancarrow's fondness for the player piano was a response to the difficulty human performers have with playing varied and changing tempos simultaneously. In his studies Nancarrow regularly dealt with complex ratios of simultaneous tempo such as that of $\sqrt{2}:2$ in *Study No. 33*. Due to the complexity of this ratio, a human performer would struggle to add any expressive nuance without it appearing as an error in timing (Kyle Gann, 2006).

In addition to robotic instruments, virtual instruments such as samplers must be considered. Samplers were only possible after advances in digital memory. Prior to this musicians utilised tape replay keyboards which stored recordings on analog tape. Of the tape replay keyboards, the Mellotron was the most popular model, being extremely prevalent in the late 1960s and 70s. The first digital sampler was the EMS Musys system, developed in 1969 by Peter Grogono and David Cockerell. Pieces composed using this instrument include Harrison Birwhistle's *Chronometer* and Hans Werner Hense's *Glass Music*. Samplers play a large part in the source material of many of the pieces in the portfolio, primarily conTimbre,[12] of which I am a developer. The use of samplers and samples is highly prevalent in modern music, much of acousmatic, electroacoustic, and popular music uses samplers and samples as a primary instrument.

However, there are some critics of this approach. One may say that the sampler imposes a rigidly defined operating system, undermining musical thought. Sample identification disrupts the listening process, and sampling divides the composer, player, and listener (Timothy Warner, 1996). However when considering the alternatives for an automatic performance medium to be either robotics, synthesiser, or physical model, the sampler appears to be the best choice for my personal aesthetic choices. Physical modelling provides the best alternative, and there are examples of commercial software achieving impressive output such as Pianoteq.[13] However, physical modelling has not advanced to such a level to provide the range of instruments and playing techniques that one may achieve using conventional samplers.

While automatic performance is possible though the means discussed above, there are some who believe that it lacks the expression attainable with a comparable human performance. Humans manage to make their performances different from that of a robot in a number of different ways, by using articulation such

[12] `http://www.contimbre.com/` accessed: 03/05/12
[13] `http://www.pianoteq.com/` accessed: 07/06/12

as staccato or legato, by making notes sharper or flatter, or expressively manipulating timbre (Miranda *et al.*, 2010). In contrast to this, performances by a robotic system are usually perceived as relatively static in expressive nuance. To combat this perceived lack of expressiveness in automated performance, systems have been developed which focus on this aspect directly. Computer systems for expressive music performance allow for automatic performances of music to be adjusted to different performance styles, and a number of these systems are outlined in Alexis Kirke and Eduardo Reck Miranda (2009). It may be said that this rigidity is fundamental to the aesthetic, as with a composer like Nancarrow, or that it is a hindrance to the musical output. However there are a number of reasons why one would want a computer to perform music expressively. For instance, one may be performing research into human expressive performance by developing computational models, developing a system for realistic playback system for a composing tool, playing data files, or creating a system for musical accompaniment tasks. This approach to automated performance is similar to the way performance is approached in this thesis. However, there is one fundamental difference. This is that for this research "expressiveness" has been emulated using probabilistic algorithms. Furthermore, the musical output included in this thesis is not attempting to emulate or replace a human performer, in that regard one may say that the concept shares more in common with the player piano works of Nancarrow.

2 Consumption and Perception of Algorithmic Music

Following this overview of the history of algorithmic music, how generative and algorithmic music are consumed in society today may be considered. To begin to determine whether the concept of a generative music media could be a valid contribution, the first thing to investigate is how society is consuming media, and how this media is perceived within society. How music is perceived by the listener has a direct influence on the main goal of producing a generative musical output. Furthermore, the method in which society is consuming media may dictate whether a software based distribution method has efficacy in real world circumstances.

2.1 Perception of Music

How a listener perceives music is extremely important for the majority of composers. There are multiple parameters controlled by automation in a generative piece, which must be musically interesting throughout. Therefore, how these parameters affect a listener's perception of a piece must be determined when considering the control methods utilised. The temporal, spatial, and gestalt[14] elements of musical perception will now be considered.

2.1.1 Temporal

Time in music can be observed on a number of scales. These are outlined in Roads (2001) as:

- *Infinite.* The ideal time span of mathematical durations such as the infinite sine waves of classical Fourier analysis
- *Supra.* A time scale beyond that of an individual composition and extending into months, years, decades, and centuries

[14] The gestalt can be described as a quality which can not be derived simply from the sum of the quantitative elements.

- *Macro*. The time scale of overall musical architecture or form, measured in minutes or hours, or in extreme cases, days

- *Meso*. Divisions of form. Groupings of sound objects into hierarchies of phrase structures of various sizes, measured in minutes or seconds

- *Sound object*. A basic unit of musical structure, generalizing the traditional concept of note to include complex and mutating sound events on a time scale ranging from a fraction of a second to several seconds

The perception of music is based largely on time. For a musician, a piece of music comprises of hierarchically ordered networks of sounds, motives, phrases, and sections. These can be looked at as time-spans, with perceptual boundaries determined by the nature of the sounds within them. From the scale above this may fit into the *Meso* category. James Tenney and Larry Polansky refer to this unit as a *temporal gestalt-unit* (James Tenney and Larry Polansky, 1980). When working with generative music one usually operates in a nested fashion, from *Macro* through to individual *Sound objects*. For example, one may "nest" the *Supra* scale within the *Infinite*, the *Macro* within the *Supra* and so on. This leads to an ordered hierarchy of algorithms operating on each temporal scale, providing musical output relevant to that particular "resolution" in time. Each of these systems can be controlled algorithmically and be interdependent. This interdependency leads to an inherent mutation of what would be a top-down hierarchy of temporal constraints. Modification of this temporal continuum can lead to changes in micro and macro structure on a per-runtime basis. These changes lead to a distortion in the perception of the piece by the listener. As sections are modified in time the overall *Macro* scale may be affected, thus the impression and character of the piece itself.

Determining the exact effect this manipulation has on the perception of a composition is difficult. However it can be analysed by looking at how one perceives temporality. Perception of music is dependent on the listener's ability to remember and recognise the material presented along with its transformations. The temporal nature of this is linear. Music unfolds gradually distorting the listener's perception of real time. This creates moments of stasis and change alternatively (Julio D'Escrivan, 1989). A listener's experience of time in musical sound was used by Husserl as a model in his analysis of time-consciousness (Joseph Smith, 1979; Douglas Bartholomew, 1985). Husserl's analysis of the temporal perspectives of melody may provide clarity upon how generative music affects the listener's perception. Husserl states that with regard to the perception of a single tone, there are three states. These are primal impression, retention, and protention:

> *The sound is given; that is, I am conscious of it as now, and I am so conscious of it 'as long as' I am conscious of any of its phases as now. But if any temporal phase (corresponding to a temporal point of the duration of a sound) is an actual now (with the exception of the beginning point), then I am conscious of a continuity of phases as 'before', and I am conscious of the whole interval of the temporal duration from the beginning-point to the now-point as an expired duration. I am not yet conscious, however, of the remaining interval of the duration. At the end-point, I am conscious of this point itself as a now-point and of the whole duration as expired (in other words, the end-point is the beginning point of a new interval of time which is no longer an interval of sound). 'During' this whole flux of consciousness, I am conscious of one and the same sound as enduring, as enduring now.* Edmund Husserl (1964) quoted in (Douglas Bartholomew, 1985, pg. 350)

Husserl distinguishes between the parts of the temporal object and the parts of the *consciousness* of that object. In the now-phase, the listener experiences the primal impression which is connected to retentions of elapsed phases and protentions pointing forward. These protentions are with held with indeterminacy, towards phases yet to come. This phase is where the operations used for the creation of generative music can lie. These indeterminacies, mirrored in stochastic processes, can be used for the composition of a musical work in real time. Applied to a melody, perception is dependent upon the intentional act. If the listener intends to perceive a motive or phrase within a melody, they will not perceive the unparsed melody; perception of the phrase will last only as long as it is present. Conversely, if the intentional act is directed toward the whole melody, the whole melody is perceived even when part of it has past. Husserl states:

The whole melody, however, appears as present so long as it still sounds, so long as the notes belonging to it, intended in one nexus of apprehensions, still sound. The melody is past only after the last note is gone. Edmund Husserl (1964) quoted in (Douglas Bartholomew, 1985, pg. 353)

Therefore, the perception of a particular melody is dependent upon the outcome of that system. If this system is in itself stochastic, then perception is dependent upon transition probabilities rather than written melody. Thus with a generative system, a listener's perception of a musical work is modeled throughout its runtime. This is important as now it can be asserted that a generative system would lead to a change in a listener's perception of a musical work on each runtime, where a static piece would not. In a static piece there are no apprehensions, no indeterminacies, and memory becomes the driving factor rather than any protention derived from perceived melody. These are the performance attributes that my generative approach to music strives to achieve.

2.1.2 Spatial

Spatial perception of sound is an important concept which should be addressed. How the listener perceives spatial attributes determines many aspects of musical representation. For example, if a group of musicians are scattered in a hall, their distribution among the audience may be regarded as more important perceptually than precise timbre and volume[15] (Trochimczyk, 2001). An example of a composer influenced by the spatial character of music was Edgard Varèse. Comments by Varèse regarding his first impression of spatial music while listening to Beethoven's *Seventh Symphony* in a concert at the Salle Pleyel reveal his thoughts:

Probably because the hall happened to be over-resonant...I became conscious of an entirely new effect produced by this familiar music. I seemed to feel the music detaching itself and projecting itself in space. I became conscious of a third dimension in the music. I call this phenomenon "sound projection"...the feeling given us by certain blocks

[15] From the listener's perspective; that is, if the listener was situated next to a large brass section, it may affect their perception of instruments on the other side of the room.

of sound. Probably I should call them beams of sound, since the feeling is akin to that aroused by beams of light sent forth by a powerful searchlight. For the ear—just as for the eye—it gives a sense of prolongation, a journey into space. Edgard Varèse (1936) quoted in (Robert Erickson, 1975; Malham, 2001, pg. 142, pg. 31)

Varèse continued his discussion on musical space in a conversation with Gunther Schuller, in which he would discuss the *projection* of musical sounds within an open space (Schuller and Varèse, 1965). There are composers that have devoted much effort into placing of the orchestra into specific spatial arrangements to achieve a "spatial" sound. The most obvious example of this is the composer Henry Brant, whose spatial arrangements of instruments were deeply integrated into his compositional thought (Harley, 1997). Brant wrote on his views of music spatiality in the article "The Uses of Antiphonal Distribution and Polyphony of Tempi in Composing" (Henry Brant, 1955). The main observations of this article are paraphrased in Harley (1997) as:

- *Spatial separation clarifies the texture*—if the music consists of several layers, "each with its own distinctive sonority scheme, over the same octave range," the presence of casually occurring unisons should be avoided by distributing the performers into widely separated positions in the hall

- *Separated groups are difficult to coordinate*—exact rhythmic simultaneities are almost impossible because of the distances between the musicians

- *Spatial separation is equivalent to the separation of textures in pitch space (if performers are together on stage)*—separation allows for the differentiation of musical strands, "with no collision or crossing of textures," and it permits a greater complexity in the music

- *Spatial arrangements must be planned exactly, but allow adjustments of details*—there is no single, optimum position for the listeners or the performers in the hall; each situation is different

These relatively practical concerns from Brant shed some light onto his use of space in compositional method. In addition to this, an example of how spatial parameters affect musical perception is approached. This is the point Brant makes that spatial separation is equivalent to pitch separation, allowing for greater complexity to be perceived more naturally. However, sounds are not often static in nature. Movement can be perceived through changes in the differences in what our two ears receive, in addition to effects on perception such as Doppler shift (Edward A. Lippman, 1963). Through artificial techniques, spatial separation of musical sound does not need to be static. Using systems based upon amplitude panning, musical sound can be moved through space. For example, the spatial work completed using dynamic systems for the *Stochastic Sampler*. It has been argued that spatialisation in music of this sort is a superficial construct, especially with regard to electroacoustic music. York Höller's argument in Brummer *et al.* (2001), was that spatialisation of music is attractive at first but quickly loses this excitement for the listener. For the experienced listener, he argues, the central content of the music is more important than any spatial qualities. Nevertheless, it is clear that the localisation of a sound source affects the listener's perception of it.

Furthermore, there is evidence that sounds, especially with regard to frequency, have a spatial character of their own. The concept that pitch has a vertical dimension has evidence in the form of experiments performed in the 1930s by Pratt, (Carroll C. Pratt, 1930) and later by Trimble (Trimble, 1934). Experiments conducted in these trials clearly found that higher tones (in pitch) are phenomenologically higher (in space) than low tones. This means that from the perspective of the test subject, the results showed that higher pitches were respectively higher in space than the lower pitches.

> *[. . .] prior to any associative addition there exists in every tone an intrinsic spatial character which leads directly to the recognition of differences in height and depth along the pitch-continuum.* Carroll C. Pratt (1930) quoted in (Robert Erickson, 1975, pg. 143-144)

This has a direct influence on the perception of all music. With regard to generative music, pitch automation therefore must have a direct effect on spatial perception of sounds in the vertical plane. Further work completed by Suzanne

K. Roffler and Robert A. Butler (1968) found that "the spatial character of the tones was even present when people who had never experienced vision were tested". In general, frequencies of higher pitch are phenomenologically higher than low ones, and with regard to timbre, bright events are higher than dark ones (Gary S. Kendall, 2010). This is an important point that follows from the discussion on time, which lead to the assertion that a generative systems could change a listener's perception of a musical work on each runtime. How this would extend to how a piece was perceived in space without any further manipulation or amplitude panning can now be considered. With the addition of systems such as amplitude panning, a listener's perception of the overall complexity of a musical work may be manipulated.

However, another consideration is how space itself may manipulate a sound. The difference between the timbre of an organ in an anechoic chamber, with that of an organ in a large cathedral can be considered. High frequency attenuation will likely occur, along with the characteristic phase colouration of reverberation (Moorer, 1979; Jon Dattorro, 1997). With regard to the perception of generative music, in this context the concentration is on the creation of artificial acoustic spaces to manipulate spatial perception in a personal listening environment. It is useful to determine some general classifications of possible spatial designs when creating such a system. These classifications are outlined in Trochimczyk (2001) in Table 2.

From an acoustic point of view, there is a clear difference between listening to music in a concert hall, and any other space such as within the home. These spaces have different reverberant qualities and thus the parameters which determine these qualities are those concentrated on when determining an artificial representation. Experiencing musical performance in a particular performance space versus another can change how even familiar music is perceived. Varèse's comments on the particularly reverberant nature of the Salle Pleyel outline this concept. In order to reproduce this feeling of varying performance space, one could incorporate a generative system to facilitate this for a personal listening

Table 2: Trochimczyk's classifications of spatial designs.

Acoustic environments	Enclosed space of the concert hall Enclosed space of any other kind Open air (different acoustic backgrounds) Variable space (mobile performers and audiences) Private, virtual space (headphones)
Sound-space types	Real sound-space (vocal-instrumental sound sources) Virtual sound-space (electroacoustic sound sources) Mixed sound-space (sound sources of both kinds)
Categories of mobility	Static performers and static audience Mobile performers with static audience Static performers with mobile audience Mobile performers and mobile audience

application. This could be achieved through a number of methods, of which two will now be considered. The first option would be to create an algorithmic reverberation system with parameters generated upon system execution. A basic example is shown in Figure 1.[16] Alternatively, one may also develop a number of impulse responses from real spaces, and feed these into a convolution unit. This would allow for composer-specified spaces to be determined either stochastically or dependent on user input. For both of these applications reverberation parameters could be determined on startup, allowing for a changing performance space on every run. It could be said that generative processes focused on spatial characteristics may determine not only the perception of the complexity of a musical work, but also the overall perception of its timbre.

2.1.3 Gestalt

How a listener perceives a the gestalt of a composition will now be considered. That is, the overall impression that it makes upon them. Listeners perceive music in relatively different ways depending on the individual. For instance, an individual's love of a musical work could be due to its structural features, whereas another individual may find that perceived emotional content results in

[16] This example is adapted from the reverb shown in `http://en.wikibooks.org/wiki/Designing_Sound_in_SuperCollider/Schroeder_reverb/` accessed: 01/06/12

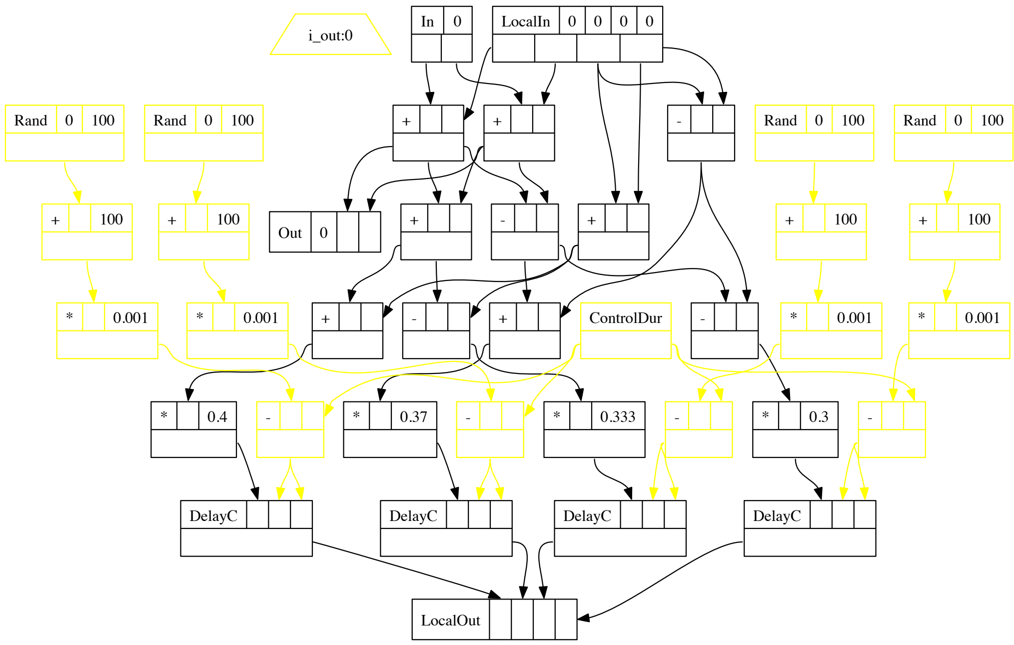


Figure 1: Schroeder reverb with randomised delay coefficients

a strong experience (Alf Gabrielsson and Siv Lindström Wik, 2003; Gunter Kreutz *et al.*, 2008). How semiotics affect how elements of music can be interpreted as meaningful to a listener will now be considered. Furthermore, how this affects their overall perception of a piece of music.

Semiotics can play a large part in a listeners perception of a piece, on the surface this may be especially apparent in experimental and electroacoustic works. For example, a listener may perceive the sound of rain in a piece of music. However, this on its own may not in itself be enough to imply any musical meaning to that sound (Simon Atkinson, 2007). In many forms of musique concrète the sound itself may not be indicative of its musical meaning and therefore may not necessarily imply a context. Nevertheless, semiotics can be useful in determining how people perceive music in some cases. Classical semiotics can be defined as a number of developments based upon Saussurean linguistic theories, and on the Peircean theoretical model of signification, which was based on a logic-based taxonomy (Gunter Kreutz *et al.*, 2008). One aspect of Peirce's semiotic lies in his categories of sign (Morag Josephine Grant, 2003). To clarify the meaning of a *sign*, James Hooper describes:

> *The difference between an idea and a sign is the heart of Peirce's Semiotic. An idea may supposedly occur in Descartes' terminology, clearly and distinctly in the mind. Because the idea is perceived introspectively in the mind, its meaning is intuited, or immediately known. A sign, as Peirce employed the term, is also a thought, but it differs from an idea in that its meaning is not self-evident. A sign receives its meaning by being interpreted by a subsequent thought or action.* Hooper (1991) quoted in (Morag Josephine Grant, 2003, pg. 176)

The most well known categories of Peirce's sign are icon, index, and symbol. These are described in Morag Josephine Grant (2003) as follows: An **icon** is a sign possessing a character that renders it significant. For example, a pencil streak which represents a geometrical line. An **index** is a sign which would lose the character which makes it significant if its object were removed, but would not lose the character if there were no interpretant. For instance, a wall with a bullet-hole in it as a sign of a shot; without the shot there would be no hole, whether anyone attributes it to a shot or not. A **symbol** is a sign which loses the character which renders it a sign if there is no interpretant. For instance, a word which signifies what it does only because it is understood to have that significance. Defining where music fits into these categories depends largely upon the music itself. Grant goes on to suggest that:

> *[...] experimental music can be understood as a shift from a basically symbolic to a basically indexical mode, and that many features of experimental music—from its focus on the social and geographical dimensions of performance through its use of various types of chance to its tendency to simplicity of materials and form—are both the reasons for, and the outward expressions of, this difference.* (Morag Josephine Grant, 2003, pg. 178-179)

Some of these points are clearly debatable, however given this proposal it can therefore be asserted that generative music may fall into the same indexical mode. This is because the music is framed within a concept of automation; presumptions on what the music is representing, the system, is of equal significance to the music itself. This does not mean it is less important in this context, it draws further attention to the music and the way it is perceived and related to. Gottfried Michael Koenig's statement on serialism highlights this concept:

> *[...] in the end it is the statistical distribution that determines the composition.* Gottfried Michael Koenig (1970) quoted in (Ames, 1987, pg. 176)

2.2 Consumption of Music

Now some aspects of how an audience perceives music have been covered, how it is consumed can be looked at. Classically, the concert hall was the place to witness a performance. However, evidence suggests that fixed media is a now the preferred method of music consumption in contemporary society. Studies have shown that the music market is now largely dominated by the personal listening audience (Tak Wing Chan and John H. Goldthorpe, 2007; Henk Roose and Alexander Vander Stichele, 2010; Noriko Manabe, 2008). Furthermore, this translates through all of the genres covered by the respective surveys, with concert attendance of classical music remaining the same even with population growth (Bonita M. Kolb, 2001). Nevertheless, a dialectical relationship may be observed between fixed media and the concert hall. One may consider that consumption of music in the concert hall and consumption on a personal level are both essential to the overall perception of music. The synthesis of these two modes of consumption leads to this relationship. Generative music in the context of this thesis attempts to provide a system which combines the particular advantages of both modes of consumption.

2.2.1 A Criticism of Fixed Media

There is one key difference between consuming a musical work in a concert hall, and in a personal environment through fixed media. This is the *experience*, with a particular emphasis on performance attributes. This is not to say that a valid experience cannot be had after listening to a recording, but that the lack of specific performance-derived attributes and a contingent nature may lead to apathy. Once the listener has memorised the nuances of even the most fantastic performance to a tee, the impact of this rendering can become dulled. Ultimately this may lead to the listener becoming bored with the material, when in reality it is the reproduction that has become boring.

> *[. . .] reproductions dominate radio, television, and the internet and rarely connect with the listener in a meaningful way.*[17] (Bhesham R. Sharma, 2006, pg. 7)

Even given a keen audience for a recording of their favourite music, after listening to the same recording multiple times the impact may be diminished through the predictability of the performance itself. This disassociation of the inherent experiential, *performance derived* qualities of music could lead to an apathetic audience, expectant of a static medium. Here parallels can be drawn to what Eduard Steuermann refers to as the "barbarism of perfection" Theodor W. Adorno (1991) quoted in (Joseph Horowitz, 1987).

> *Perfect, immaculate performance in the latest style preserves the work at the price of its definitive reification. It presents it as already complete from the very first note. The performance sounds like its own phonograph record. The dynamic is so predetermined that there are no longer any tensions at all. The contradictions of the music material are so inexorably resolved in the moment of sound that it never arrives at the synthesis, the self-production of the work, which reveals the meaning of every [Beethoven] symphony. What is the point of the symphonic effort when the material on which that effort was to be tested has already been ground up? The protective fixation of the works leads to its destruction, for its unity is realised in precisely that spontaneity which is sacrificed to the fixation.* (Theodor W. Adorno, 1991, pg. 43)

Whereas previously, as noted by Karl Marx: "The service a singer performs for me, satisfies my aesthetic need, but what I consume exists only in an action inseparable from the singer, and as soon as the singing is over, so too is my consumption" (Mark Katz, 2004, pg. 13). Now, recordings last forever, consumption is separated from the action of performance. This concern of eternal reproduction

[17] My understanding is that this comment is largely directed toward the depersonalisation of music within media. For example, if one is subjected to the climax of a particularly meaningful piece of music every time an advertisement appears on the television, the impact of this music could likely be skewed by the connotation.

was highlighted by Busoni in 1919, when he wrote of recording that "to do it is stupid and a strain". "Not letting oneself go for fear of inaccuracies and being conscious the whole time that every note was going to be there for eternity; how can there be any question of inspiration, freedom, swing or poetry?" quoted from (Joseph Horowitz, 1987, pg. 415). A generative, dynamic approach to the replication of musical concepts would relieve the composer from the fear that every note would be there for eternity. Each time the composition is to be listened to it would be a new and unique "performance" rather than a static phonograph of a single past performance. This reintroduction of the concept of "performance" to the personal listener would allow for musical experience to find its way back into the realm of personal listening.

Walter Benjamin, Adorno's friend, characterised aesthetic secularisation as the loss of the 'aura' of the work of art. The work of art before mechanical reproduction in Benjamin's account, was a symbol of divine transcendence; in the modern era it was the concern of a specialised branch of culture and an object of conscious engineering (Gary Zabel, 1989). According to Benjamin, the uniqueness of a work of art is identical with its embeddedness in the context of tradition. He makes the analogy of a classical statue of Venus, which occupied a context of worship for the Greeks, but a threatening idol for medieval clerics. Each of these groups were struck by the *singularity*, the 'aura'. Benjamin asserts that it is crucially important that this aura is never completely separated from its ritual function.

> *The 'one-of-a-kind' value of the 'genuine' work of art has its underpinnings in the ritual in which it had its original, initial utility value.* (Walter Benjamin, 2008, pg. 11)

Benjamin found the loss of aura primarily in mass-produced art such as photography, cinema, newspaper and journalism. One of his main points was that the fact that a work of art can now be reproduced by technological means alters the relationship of the mass to art. "The more that the social significance of an art diminishes, the greater the extent to which the critical and pleasure-seeking stances of the public diverge". Mass-production leads to *getting used to* perceptual apparatus, the consumption becomes habitual, achievable in a state of distraction. This reception in a state of distraction is apparent in all fields of art and is clearly symptomatic in popular musical media. Static reproduction hinders value by persuading the audience to adopt an appraising, but inattentive stance toward artwork.

> *The audience is an examiner, but a distracted one.* (Walter Benjamin, 2008, pg. 35)

When the listener is expectant of a specific dynamic, a perfect reproduction and a static entity, they become inattentive, distracted. The introduction of generative algorithms to the creation of musical media would demand attentiveness from the listener. For example, in a similar sense, a performance of Stockhausen's *Aus den Sieben Tagen* should not be identical every time, as I am sure it was not intended to be. The essence of the piece is in the interpretation of the score, rather than the musical output itself. Creation of music based on a generative algorithmic framework leads to a media which embraces the temporal, contingent nature of music and reintroduces an 'aura' to each reproduction.

To further develop this postulation, how the recorded media has changed the way people actively listen to music itself can be considered. Indeed, even classical music has been directly influenced by the imposed constraints of recording. For example, Igor Stravisnky's *Sérénade en LA pour Piano* was written so that each of the four movements would fit the three-minute limit of a ten-inch, 78-rpm record side (Mark Katz, 2004). However these constraints need not imply a limitation, constraints are applied in many forms to musical composition. This could be in the form of harmony, tempo, pitches utilised, or playing style. In contrast to this however, recorded media may have a direct influence on a whole genre or region of music. Its influence can lead to homogenisation or variation within a whole subset of musical landscape. Two such examples of this are found in North Indian classical music and the gamelan music of Java.[18]

From a social standing, recorded media has allowed a much wider range of people to consume and be influenced by music. A concert ticket may be of much

[18] These examples are covered in depth in Mark Katz (2004)

higher value than its recorded counterpart, and with free radio services music can be delivered to the masses. In addition to this, a concert ticket is only valid for one "consumption" whereas the recorded media is available to listen to for the rest of eternity, as long as it is kept in good working order. It represents a tangible commodity which can be utilised at any point (digital media is an exception to this as it does not occupy any tangible space). A generative approach (dependent on delivery method) would allow for the experiential advantages of a concert performance, along with the commodity value of a fixed medium.

2.3 Context in Current Society

This chapter has covered some concepts that may be observed as relatively theoretical, removed from the listener to some degree. To provide some context, how some of the concepts approached above can be contextualised within current society will be considered. To do this, two things must be approached: how such a medium can be delivered,[19] and how this idea of generative music delivery may find a context within society.

2.3.1 Mediums for Musical Delivery

Methods of musical media delivery have changed radically in the past 50 years with the onset of CD, MP3, and now evidence for a move toward the 'app' (Anna Sophie Christiansen, 2001; Reebee Garofalo, 1999; Thor Magnusson, 2011). One example of the change in musical consumption is 'Internet music'. Music and the internet is a relatively recent occurrence, with bandwidth only becoming usable in the past decade. Nevertheless the internet is now entwined in music, in both production, consumption, and distribution (Steve Jones, 2000). Music production and the internet can be broken into certain general types, defined in Andrew Hugill (2005a) as: music that uses the network to connect physical spaces or instruments; music that is created or performed in virtual environments or uses virtual instruments; music that translates into sound aspects of the network itself; music that uses the Internet to enable collaborative composition or performance;

[19] This is covered in much more technical depth in section 4.2.2

music that is delivered via the Internet, with varying degrees of user interactivity. Specific examples of Hugill's involvement with internet music are contained in Andrew Hugill (2005b). Other examples of sonic works for the Internet and computer networks are outlined in Peter Traub (2005). However with respect to the work contained in this thesis, the Internet is approached more as a tool for distribution rather than a data source. The generative systems contained within the portfolio are self-contained systems designed to run on a standard system with few third-party dependencies. This means that the systems may be released on any media which supports executable content. For example, a CD may instead of playing in a media player, automatically run a binary file which generates the music live. Through development, this evolved into a system based around internet-radio based delivery, discussed further in section 4.2.2. One important aspect of the Internet in respect to music is its social potential. As the focus is upon music for personal listening, this will now be considered.

2.3.2 Generative Music; Social Context

Throughout the research period, during the composition process, it became clear to me that the generative music being created had no consonant stage for delivery. It was suggested to me that music should have a *social context*. For example, a concert hall is a coherent social construct wherein people go to enjoy music within society. In addition to this, music released for concert hall consumption has a clear audience within that space. Therefore, the music created for this research project should have a social context, and a feasible delivery method with the ability to reach a wide audience effectively. With regard to the audience, the focus for the work accompanying this thesis has been upon the personal listener. This is in fact a huge audience and as discussed previously, contains the overwhelming majority of music listeners. However, this focus raises important questions about the social potential of the music produced. When there are no concert performances, it is difficult to place a piece of music within a social context. A question can be raised that without a physical social presence, perhaps no-one would ever hear or find out about such music. Therefore generative musical content requires a convincing stage for social consumption that provides a social context to the work. Two examples of such platforms are the Internet, and the gallery installation.

Perhaps the most applicable medium for the distribution and social presence of such content is the Internet. Communities of computer-based generative artists have emerged online, such as `http://www.generative.net/` and the associated *eu-gene* mailing list. Further examples of this kind of group can be found within forums dedicated to real time software environments. These communities, focused upon environments such as MaxMSP, PureData, and SuperCollider, have large numbers of people working within a similar framework. This is further expanded through social media and the sharing of musical works. For example, groups on SoundCloud.[20]

In addition to the Internet, the gallery is a relatively consonant stage for generative media. This raises new compositional potential as the audience for an installation is not fixed in seats. For instance, temporality in the context of installation performance may be approached in a completely different manner. Pieces which last indefinitely are possible, for example complex recursive algorithms can move into the macrostructure leading to extended repeated forms. One example of such extended form is Jem Finer's *Longplayer*,[21] further examples can be found in Nick Collins (2002). In addition to this, audience interactivity becomes a realistic parameter for musical application. Members of a seated audience in a concert hall provide little data for an interactive system. However if the audience is free to move then their movement is a quantifiable parameter that can potentially be applied to a facet of the musical work.

2.3.3 Generative Music in Concert Setting

The focus of this investigation is primarily concerned with the personal listening audience. However, as the concert is the traditional avenue for classical music consumption, the possible ways a system for generative reproduction of music could

[20] Two examples of this are the SuperCollider group: `http://soundcloud.com/groups/supercollider/` and the MaxMSP group: `http://soundcloud.com/groups/max-msp-users/`. A more specialised example of such a site is: `http://www.sccode.org/`, where people may share SuperCollider code with each other in a similar fashion.

[21] `http://www.longplayer.org/` accessed: 10/07/12

be translated to this context will now be considered. These two approaches offer fundamentally different aesthetic output. For concert performance my personal aesthetic preference would be biased toward a human performer. However, there are possibilities for compromise in this situation through human-interface-devices and performance parameter control.

One approach that could be taken in the context of concert performance would be using automatic performers. This approach would enlist robots for performance of generative works in a live context, such as in a concert hall or installation space. Indeed, some of the work for this portfolio may be represented in this fashion instead of samplers. The works for automatic piano are possibly the easiest to realise in this fashion, and a number of them were recorded using this performance method. By utilising robotics as a performance medium we are tackling the concern that in a concert setting, the inanimate nature of a sampler could affect audience perception of a piece in a negative way. The argument is that as there is an inherent lack of visual connection in a sampled piece, the experience is dulled in some fashion. Robotics would therefore allow for the visual and physical cues in performance to be retained while also keeping the fundamentally "automatic" performance medium.

However, one possible issue with the use of robotics is an extension upon this problem: some of the human expression of performance is sacrificed. Developing this argument, it can be (bluntly) asserted that with regard to concert hall performance, the machine musician is quite literally Adorno's "flawlessly functioning, metallically brilliant apparatus, in which all the cogwheels mesh so perfectly that not the slightest hole remains open for the meaning of the whole" (Theodor W. Adorno, 1991). However, this assertion completely disregards aesthetic nuance introduced by the robotics themselves, and any compositional decision that would have lead to their use. One well-known composer who utilised robotics almost exclusively was described by György Ligeti as "the greatest discovery since Webern and Ives... something great and important for all music history! His music is so utterly original, enjoyable, perfectly constructed, but at the same time emotional... for me it's the best music of any composer living today" (Kyle

Gann, 2006, pg. 2). Conlon Nancarrow in fact singled out, thematized, and turned the player pianos deficiencies to his aesthetic advantage. The inability of the player piano to convey "the human element of personal expression" (Henry Cowell, 1996) was played upon and in some cases even developed (Drott, 2004). For example, the inexpressive timbre was accentuated through placing tacks and strips of leather on his pianos to "achieve greater rhythmic clarity and more incisive attacks" (Philip Carlsen, 1988). In this case the modified, metallic sound of the instruments in fact asserts their mechanical nature, and their status as musical automata. The music of Nancarrow clearly was written *for* the automaton, rather than merely performed by it. Furthermore, Nancarrow's music constitutes proof that 'aura' or 'feeling' in music is certainly not constrained to the context of human performance.

> *[...] ever since I'd been writing music I was dreaming of getting rid of the performers.* quoted in Charles Amirkhanian taken from (Drott, 2004, pg. 534)

A compromise could be made here for the clear context of concert performance. In this context one could utilise a live performer with a human interface device modifying parameters sent to the robotic performers. This has been done previously in works such as Karlheinz Essl's performance[22] of his *Lexikon-Sonate*. In this performance he "plays" the piano using a number of faders, and a laptop controlling the algorithms which generate the piece. This performance context allows for human expression to merge with automatic performance and generative music, leading to a symbiosis of the two styles. However, one practical difficulty with this is that in a live-electronics setting one would require a partner to monitor the output of the loudspeaker systems as they would be in front of the stage. In essence a system such as this would retain the "pure" automatic performance aesthetic, while providing humanistic expression and worthwhile visual and audio cues for a live audience.

Robotics are however, not the only way to approach this issue. A technique which bypasses any of the debatable aesthetic issues which may arise from auto-

[22] `http://youtu.be/aOOTafrusbw/` accessed: 06/07/12

matic performance is that of real time music notation. Real time music notation can be placed within the framework of algorithmic and computer assisted composition. In the context of generative music, it can be asserted that there is a dichotomy between human and machine musicianship. Samplers, robotics, and other forms of automated performance can certainly lack human expression in this context. This is especially apparent in any standard concert setting. Score generation would allow for human nuance in performance while also retaining formalised generative output. Real time music notation systems would wait to create the score until during the performance, producing dynamic musical score that may contain either conventional notation (David Psenicka, 2009; William A. Burnson and Hans G. Kaper, 2010) or graphical representations. Score generation techniques allow for an open-form aesthetic, in which the musical score is interpreted differently in each performance of a composition. However, they have some limitations. For a completely formalised system utilising standard notation, real time notation has specific limitations regarding the complexity of the score. If a performer is to read the score in real time, the composer must consider the complexity of the output in order to retain a fluid interpretation. This limitation can be overcome in part by allowing for a semi-improvised form. For example, *Champ d'Action* (1998) by Karlheinz Essl. In this piece the automatic score shown to the performer parameterises performance into six elements. These are phrases, pauses, register, sound, tendency, and speed. Through these six parameters and a further written description, the performer is to interpret their version of the score (Jason Freeman, 2008). This use of relatively *fuzzy*, qualitative algorithmic output bypasses any of the problems of reading complex score on the fly while capitalising on the performers own musicianship.

3 Interactive Systems: *Stochastic Sampler*

From contextual and theoretical considerations, the discussion will now move toward more technical aspects of the investigation. Realtime systems developed for this portfolio concentrate primarily upon the individual listener, rather than a group of listeners as would be found in a conventional concert audience. Furthermore, the software developed for this task is autonomous and non-interactive during playback. However, during the course of research other approaches were investigated. For installation and live performance of generative music, an interactive system was developed. This was one of the first projects embarked upon and provides a good background to the signal processing and generative music systems that followed it.

To provide some context to the investigation, the musical roles in which an interactive system could operate in a performance setting should be considered. For example, an interactive system designed to provide some "improvisatory" musical feedback, a human performer would play alongside a computer. The computer performer would take data from the performance of the human performer, parse it, then produce coherent musical output as a response to the stimuli provided. The *Stochastic Sampler* attempts to operate in this way, listening to an audio stream from a performer, parsing the data for musical parameters, and outputting its response based on an algorithm. Examples of previous systems of a similar type are George Lewis' *Voyager* (Lewis, 1999) and Robert Rowe's *Cypher* (Rowe, 2001). The *Stochastic Sampler* is an example of a generative system which applies basic machine learning techniques to provide some interactivity.

Another possible context is that of a modular audio effect, a parallel would be with an effects pedal. The *Stochastic Sampler* focuses on granular synthesis as the basis for its output. A background to this technique is covered in depth in section 3.3.1. To allow for this application, the system was developed with modularity and efficiency in mind. However, the context for performance focused mainly upon installation, or improvised performance with accompaniment. As an expandable interactive system which operates on a standalone basis, the

Stochastic Sampler is able to provide a dense output with no further augmentation necessary. Applied in the context of installation it would operate in the same way it would in a performance context, based upon the data it was provided with.

Expandability and modularity were important considerations when designing the system. How these goals were approached, and the technical aspects of the system and the way it was designed will now be looked at.

3.1 Concept

The overall concept for the *Stochastic Sampler* was initially born from studying the works of Iannis Xenakis and Conlon Nancarrow. This was coupled with an interest for signal processing, improvisation, and live performance. It can be defined as a modular system comprising two basic elements, a control module and a player module. This is a similar macrostructure to previous systems such as Robert Rowe's *Cypher*, and George Lewis' *Voyager*. The control module deals with parsing the data though a number of parallel Markov chains, the player module operates sampling and playback. The system itself originally operated within the MaxMSP environment, and was initially designed to operate on its own as a standalone application. This first iteration of the system will now be looked at. This includes a consideration of the two major modules, and how the system was designed with reference to the goal of creating an interactive system.

3.2 Control Method

Perhaps the most important module, the control method determines what information will be passed to the player module. In essence this aspect of the system decides which musical information is worth storing, and which information will control the musical output. The control module was designed with the idea of musical relevance in mind. Ideally, when a performer works with the system it should output musically coherent information. This was attempted by taking data for amplitude, pitch, and duration from the audio input. The data then could be collated and parsed. Here the influence was taken from Iannis Xenakis, and Markov chains were used as the primary control algorithm. The reasons for this choice of algorithm will now be considered.

3.2.1 Markov chains

Markov chains were well suited to the task and data of the *Stochastic Sampler*. A Markov chain models the behaviour of a sequence of events, each of which can assume a fixed state within a finite range. Changes between these states are known as transitions (Ames, 1989). The behaviour of a chain is approached as a set of numbers known as transition probabilities. Each of these transition probabilities determine the likelihood of the chain jumping to a particular state. Therefore, with respect to music, each state would refer to a particular element within a musical parameter. For example one amplitude state may be pianissimo, and one may be fortissimo. The transition probability between pianissimo and fortissimo would determine the likelihood of this happening. With regard to the *Stochastic Sampler*, the more times a performer made the transition from fortissimo to pianissimo, the larger the transition probability would become. Therefore the frequency of this occurring in the system's output would increase.

Markov chains have been utilised in music since Lejaren Hiller and Leonard Isaacson employed them in their *Illiac Suite* (1956). Further work was done by the composer Iannis Xenakis, who used simultaneous chains in the production of *Analogique A* (1958), *Analogique B* (1959), and *Syrmos* (1959). Xenakis' approach to Markov chains was to utilise them in parallel, creating "screens" of musical space which he referred to as "grains" of sound (Xenakis, 1992). A similar approach was taken when conceiving the idea for this system, where each voice of the system has its own chain which determines its output individually.

Markov chains were chosen as the basis for the control module as they can simply and effectively parse the data given to them in a musically relevant way. This is however largely dependent on the order of the chain, in essence the number of preceding elements referred to when choosing a new element for output. Originally the system was developed with a first-order Markov chain implementation. This was then increased to fourth-order when the system was ported to Java, and then to nth order in the SuperCollider port. As the order of the Markov chain increases, its output can be viewed as becoming more relevant to the data it was supplied with. For this application, this should hopefully increase musical coherence between the performer and the system.

3.2.2 Application

The system applies Markov chains to both the duration and pitch parameters of the incoming data. Transition matrices are stored, and the output of these is abstracted in order to allow for multiple voices. The user may increase the number of voices to their individual specifications. This parallel Markov chain application allows for an increased level of variation in the output. Technical aspects of the implementation of this system will now be considered.

Initially, data must be collected from the audio stream and turned into something that can be parsed by the Markov chain system. To collect the data for the system to operate, the pitch following objects `sigmund~` and `fiddle~` were utilised (Miller S. Puckette *et al.*, 1998). These objects gave access to the estimated pitch of the player along with the duration between attacks, without requiring for a MIDI input of any kind. Due to the nature of this detection method there are some inaccuracies. However, these can be reduced through some sanitisation of the output. For example, one possibility would be to round to the closest MIDI note instead of allowing for microtonal inflections. This can be achieved by simply casting the floating point output of the pitch-tracking object to an integer after it is converted to a MIDI note value. In part the inaccuracies were considered in the development as adding "character" to the system, much like Lewis' application of random number generators to his system *Voyager* (Lewis, 1999). Certain arguments given to the pitch tracker can change the output wildly. For instance, if the re-attack time is modified then the object will detect many more or less notes. The re-attack parameter takes two arguments: threshold, and duration. This means if the strength of a pitch changes by x dB within t ms, the object will output a new note. Due to background noise this can have the affect of adding spurious input, mutating the transition tables probabilities and increasing the overall density of output.

Once the data is converted from audio to numerical data, it is sent to the Markov chain system. The first iteration of the system was realised in MaxMSP. The relatively simple method of creating a Markov chain was greatly useful in initially creating the system. A patch was created which applied this first-order Markov chain system to the pitch and duration parameters. This could then be coupled with the player module and abstracted, allowing for multiple voices to be added to the system. When the system was completed it comprised of fifty simultaneous voices. In order to retain some sort of musicality with this number of voices a parameter for density was added to the final machine. This parameter allowed for the duration parameter to be multiplied by any chosen value. As this duration is responsible for the triggering of the player module, this control allowed the user to immediately modify the density of the output. This allows for the relative difference between performer derived durations to be retained, while modifying the output sound. For example, if the multiplication value is set to two, all of the durations will be doubled, halving the density of the output.

Now the system was able to operate interactively, and use performer generated data to create output. However one issue was that it could not generate any data without stimulation from the player, the triggers for the system were completely external. Therefore, to allow the system to operate without stimuli from the performer and generate its own music, a feedback loop was applied in the form of a gate. This means the system can (if required) feed back into itself and provide its own triggers. However, this can have the effect of creating new entries in the probability tables. Due to this manipulation of previous probability data, the performer derived data is mutated with phrases generated by itself. This creates a dichotomy between the "mechanisation" of the performer generated material, and the "humanisation" of the computer generated material through random number generators. Here there is a comparable element with Lewis' *Voyager*. Similarly, he allowed his system to perform without stimuli. He states "the computers own musical behaviour is the product of its own initiatives and its response to outside input when the program has determined that such input is present" (Lewis, 1999).

The system now provided interactive output, along with the ability to provide its own output without any performer stimuli. However in the initial stages of playback some inconsistencies and problems were noticed in the output. This is because when the system is in the initial learning stage, there are relatively few options for it to take. It will regularly get stuck without any new states to go to, reset itself, and end up ultimately repeating itself. Systems such as Lewis' *Voyager* also faced this challenge. Lewis states that a follower-leader system was not implemented in *Voyager*; in order to get the system to play in a specific way, the performer had to perform like that. As the system was based largely on data it received, the performer had to allow it to learn how to react to stimuli if a specific musical direction was intended by the performer (Lewis, 1999). To combat this, later versions of the *Stochastic Sampler* allowed for a predetermined "brain" to be loaded into the system before initialisation. This meant that it could be tuned before a performance to act in a specific way. For instance, a transition matrix may be loaded into the system which only contains notes within a specific key. When the system is started it would work within this finite range of notes until the performer wished to change. The advantage of this is that the learning process starts from a much more advanced position, allowing for coherent output from initialisation.

3.3 Player Method

The control methodology and how the data is parsed and manipulated has now been covered. However, what this data will do to the audio output must be considered. The player module determines what the audio output is, and how the control method's numerical data manipulates this output. When deciding on the method of synthesis to utilise for the player module in this system, the main question was that of timbral consistency. If a performer was working with the system, the output of the system must work with the player and not be too dissonant timbrally. It should augment the output of the player or players, and be consonant with their overall sound.

After considering this, it was decided that the best, most consonant timbre to utilise was that of the performer's instrument itself. In order to fit with this criteria and the modular design of the control system, the player method would have to be a contained sampling and playback system in its own right. The player module takes the messages given to it from the control module, and transposes the samples stored in its memory by the correct ratio for playback. The player method of the *Stochastic Sampler* truly encompasses the *stochastic*: many of the operations it performs are based upon not only the Markov control signals, but also probabilistic gating methods. Overall the system is based upon the tenets of granular synthesis, the reasons this was chosen as a primary output approach is discussed below.

3.3.1 Granular Synthesis

First a short background to granular synthesis will be looked at. This should hopefully be useful to determine some of the compositional applications of the system, and to highlight some of the advantages of such an approach. Granular synthesis was first automated in a non-realtime environment by Curtis Roads in 1975 (Roads, 1978). The technique had been presented in the article "Acoustical Quanta and the Theory of Hearing" by Gabor (1947). A compositional application of grains was examined in Iannis Xenakis' 1971 book Formalised Music (Xenakis, 1992). Before this, Xenakis had given an instrumental musical example of granular synthesis in his piece *Pithoprakta* (1956). Xenakis' tape piece *Concret PH* (1958), comprised of layers of recordings of burning charcoal also demonstrates a granular, self-similar approach to sound (Scipio, 1998). Roads' original application of this technique was a system designed as a front end for a MUSIC V installation (Roads, 1978). Granular synthesis was then adapted to a realtime environment by Barry Truax, who employed the DMX-1000 Digital Signal Processor to take the computational burden from the computer itself (Truax, 1988). Granular synthesis blurs the line between micro and macro structure in compositional technique. The densities that can be utilised by a composer using this technique range from low level tempos through to soundscape-like textures.

A standard implementation of granular synthesis may incorporate controls for amplitude, frequency, and grain duration parameters. These may be controlled manually or through deterministic or non-deterministic methods that fluctuate between predetermined boundary values. Some prior methods of control for this technique include Barry Truax's method, which utilises a hierarchy of control levels (Truax, 1988, 1990). These are based upon score files, ramp files, and tendency masks. By using this kind of control structure, Truax was able to control a large number of events at any one moment, and was able to create complex output from a minimal amount of source material. Due to the large amount of events requiring control when utilising this method of synthesis, a system which can operate at the required speed must be considered. This highlights the efficiency necessary in a system for complex granular synthesis application.

With regard to the *Stochastic Sampler*, granular synthesis provided a perfect way in order to be able to work within a wide range of output density. The system would allow for large samples of multiple seconds, or extremely small millisecond grains to be played. Allowing for large samples to be taken meant phrases could be captured as single elements. These phrases could then be rearranged and transposed, creating new musical statements from previous material. In addition to this, very small millisecond length nuances may be captured and played back at a much faster tempo, creating a new texture. These two extremes and the densities between them would create a large range of possible timbres, and ideally an interesting output. The transposition of these larger phrases and shorter grains forms one of the major compositional approaches developed throughout this research. This is considered further below.

3.3.2 Application

Now a background has been covered, how granular synthesis was applied to the system's player module will be looked at. The core of the player system was first developed in MaxMSP as a probability controlled module for sampling and playback. Instead of taking a third party granular synthesis module the system was written from the ground up. This was in order to have full control over the parameters, control method, and functionality. At its core the module takes thirteen arguments: audio in, pitch (in), pitch (out), volume, beat detection in, sample length, frequency of sampling, attack, decay, sustain, release, scale, and glissandi envelope duration. The player module was abstracted, meaning each voice would have individual values for these arguments. This was achieved through a software control patch which may be controlled with a midi controller. The values for attack, decay, sustain, and release are automatically modified so they do not exceed the current sample length.

Transposition of the sampled sound is done through playback rate scaling.[23] This was decided upon due to the increase in efficiency over more complex methods, and also the affect it would have on the tempo of the recorded samples. The concept was to create an efficient system with no FFT smearing or granular artefacts, and a complex output in both the frequency and time domains. After studying the works of Henry Cowell (Henry Cowell, 1996) and Conlon Nancarrow (Kyle Gann, 2006), this idea of playback rate transposition became more appealing. By transposing by the pitch decided upon from the Markov model, the tempo is scaled by the same amount. This leads to a layered output comprising of multiple tempos joined by the harmonic range of the performers input. The method is one of the core compositional ideas used throughout this thesis, and this original application in the *Stochastic Sampler* was key to the development of the overall aesthetic. The concept is described in Cowell's *New Musical Resources*, where he describes the technique:

> *Rhythm presents many interesting problems, few of which have have been clearly formulated. Here, however, only one general idea will be dealt with—namely, that of the relationship of rhythm to sound-vibration and, through this relationship and the application of overtone ratios, the building of ordered systems of harmony and counterpoint in rhythm, which have an exact relationship to tonal harmony and counterpoint.* (Henry Cowell, 1996, pg. 46)

To further develop the functionality and musical possibilities, microtonal scale playback was added to the system. This is achieved within the function for the

[23] Also known as sample rate conversion

derivation of sample playback ratio. As shown in this example, the ratio can be found from:

$$r = (2^{\frac{1}{n}})^p = (\sqrt[n]{2})^p$$

Where r represents the playback speed ratio, p represents the pitch (MIDI), and n represents the number of notes in one octave. The original patch stores the pitch detected when the sample was recorded within a specific period of time. This allows the difference to be calculated, and the correct Markov model derived transposition to be applied to the recorded sample. If for any reason the data is compromised and a zero appears, the sample is played back at its original speed with no transposition. As the player module has both sampling and playback functionality, the question arises of when the module should play and when it should record. Some consideration has to also be made for the difference in duration between one sample and the next, and the difference in the playback duration when transposing. The decision for playback or record functionality is based upon a simple user controllable probability parameter. This has a knock-on effect of modifying the density of the output. If there is a 70% probability that the sampler will record new data rather than playing back a previous sample, the chance of playback is only 30%. This reduced output density is coupled with more varied output, as the system changes the sample more often.

3.4 Java

In the introduction to this chapter, use of the system in a modular fashion in a signal processing chain was touched upon as a possible context for application. This would use the system in a similar fashion to a guitar pedal, or similar effects module in a chain. However, the original realisation of the system in MaxMSP had some efficiency issues meaning it was unable to operate as a modular part of a signal chain effectively. To increase efficiency the system was ported to Java through the use of the `mxj~` API. This allows MaxMSP to spawn its own Java Virtual Machine, and Java code to run within this environment within MaxMSP. Instead of writing in C++ which would be the obvious choice for efficiency, Java was preferred. This was mainly due to the more streamlined development process and cross-platform capabilities of Java. Furthermore, a number of APIs were used which were specific to Java and enabled more effective use of development time. This Java version had advantages for both the player and control modules of the system, which were both rewritten.

The player module in Java was designed as a single voice module, for use within a `poly∼` object in MaxMSP. This would mean the number of voices could be changed extremely simply through the functionality of `poly∼`. In addition to this, the `poly∼` object allows for CPU multithreading, increasing the efficiency of the system radically in multi-core systems. By coupling the Java based sampler with multithreading support, a massive increase in efficiency over the original patch was attained. This would ultimately allow for more voices, or for the system to be utilised within a signal processing chain. The control module was written in Java using the *jMegaHal*[24] API. Originally *jMegaHal* was intended for use as a conversational robot on internet relay chat, however it works perfectly for the requirements of this system. This API allowed for an extremely quick development process, and allowed for a fourth-order Markov chain to be used within an efficient environment. This also allowed for the functionality of predetermined "brains" to be given to the system before it was instantiated. This allows for instantly coherent output rather than the "learning" period while the system gathers musical data.

There were some lower level advantages to the Java approach. One example of this is per-sample control, and thus control over the interpolation methods utilised. Multiple interpolation methods were tested to determine which had the highest quality output while retaining efficiency. The interpolation method was to be applied to both the amplitude envelope and sample transposition function. Functions tested ranged from standard linear interpolation to four-point Lagrange interpolation. A balance between efficiency and quality was found with the 4-point 3rd order (X-Form) Hermite algorithm, sourced from Olli Niemitalo (2001). For reference, the coefficient matrix for this interpolation algorithm is:

[24] `http://www.jibble.org/jmegahal/` accessed: 09/09/12

$$X = \begin{pmatrix} 0 & 1 & 0 & 0 \\ \frac{-1}{2} & 0 & \frac{1}{2} & 0 \\ 1 & \frac{-5}{2} & 2 & \frac{-1}{2} \\ \frac{-1}{2} & \frac{3}{2} & \frac{-3}{2} & \frac{1}{2} \end{pmatrix}$$

Where t is the fractional read-index, and *x0* - *x3* are the four interpolation points, this can be translated into Java as:

```
double c0 = x1;
double c1 = .5F * (x2 - x0);
double c2 = x0 - (2.5F * x1) + (2 * x2) - (.5F * x3);
double c3 = c3 - (.5F * (x3 - x0)) + (1.5F * (x1 - x2));
return (((((c3 * t) + c2) * t) + c1) * t) + c0;
```

This balance of quality and efficiency is extremely important in a realtime system, which is why it is covered in relatively precise detail. Sample interpolation represents a task which utilises many CPU cycles, therefore this balance was found by optimising this element. By concentrating on elements such as this, the efficiency of the system was increased to the point that it was usable in a modular context. As it is designed around the `poly~` object in MaxMSP the number of voices can be modified dynamically to suit the efficiency requirements of the system.

3.5 Further Development

The framework for the *Stochastic Sampler* was a large stepping stone in the development of my personal signal processing ability. Due to this, it became a base for further experimentation in algorithmic composition. Let us now discuss some of the further work that was done on top of this base element.

3.5.1 *Stochastic Sampler 2* (2012)

First, the improvements made to the original system must be considered. Initially, the *Stochastic Sampler* and all composition framework was centred around the MaxMSP environment. However, with the creation of the compositional methodology in SuperCollider this became largely incompatible. Furthermore, the systems created in MaxMSP/Java had many idiosyncrasies which rendered them very difficult to use in a creative context. Therefore, a SuperCollider rewrite of the system was developed in 2012. This rewrite coupled efficiency with ease of use, and complete integration with the current composition methodology.

There are some fundamental differences in design between the original system and the *Stochastic Sampler 2*. At the core of the system, the Markov implementation[25] allows for an nth-order approach. As the system was written as a SuperCollider class, this means at any one time there may be many *Stochastic Sampler 2* objects, with different order Markov chains. In addition to this, the number of parallel Markov chains for each voice was increased to include the amplitude parameter. Each voice has three Markov chains of nth-order, operating upon duration, pitch, and amplitude parameters of the incoming audio stream. This allows for an output with much more musicality.

Most importantly, this implementation allows for the system to integrate fully with the current compositional framework. In addition to this, by writing as a SuperCollider class further development of the system is trivial. This allows for the system to grow with the compositional methodology, and for further functionality to be added without major issues. Furthermore, due to the hugely improved efficiency the applications of the system are widened. It can operate as a system within a signal processing chain, or as a standalone system for interactive performance/installation. It has been tested with voice-counts of up to 300 at a 48kHz samplerate, with no perceivable artefacts.[26] This amounts to 900 parallel nth-order Markov chain systems, and approximately 50 minutes of sample data.[27]

[25] This is provided by the MathLib quark.
[26] 2010 MacBook Pro i7-620M, 8GB RAM
[27] Sample length is a user defined variable.

3.5.2 *CA Sampler* (2010)

To this point, Markov chains have been looked at as the primary avenue of investigation into methods for granular synthesis using the sampling tools developed. However, due to the separation of control and player modules other techniques could be investigated using the pre-existing sampling system. The major element of further investigation in this regard was the application of Cellular Automata to granular synthesis. This was done within MaxMSP and Java in the same way that the *Stochastic Sampler* was realised. The background to the topic will now be considered, in order to determine why such systems were considered for this application.

Cellular automata have starting points far back in the sciences, but were first formally introduced in the 1940s (Chareyon, 1990). They were first conceived by Stanislaw Ulam and John von Neumann in an effort to study the process of reproduction and growths of form (Dave Burraston and Ernest Edmonds, 2005). In some ways it is possible to say Pascal's triangle is the first cellular automaton (Peitgen *et al.*, 2004). Cellular automata are perfect feedback machines, finite state machines which change the state of their cells step-by-step. A cellular automaton can be seen as a grid of cells which represent a state at a given moment. As the time progresses, every cell takes on a new state depending on the previous state, and the state of the cells adjacent to it (Chareyon, 1990). To run a cellular automaton a number of things are required: an initial state of its cells, and a set of rules. These rules describe how the state of the cells in the next step are determined from the states of the cells from the preceding step. The most common examples of cellular automata are two-dimensional and rectangular, or one-dimensional and linear. In the case of a one-dimensional automaton, the cells lie on an unbounded line or one which wraps around. The number of states is usually represented by an integer n. Every state is represented by an integer from $0 - n - 1$ and may be expressed graphically with different colours. Common neighbourhoods for linear automata are made of three cells (a cell and those touching it on either side), five cells (a cell and the two nearest on either side), or more commonly of the $2n + 1$ cells centred on the current cell (Chareyon, 1990).

One of the most popular examples of a cellular automaton is John Conway's "Game of Life". This is a model for the evolution of a community whose members occupy cells on a rectangular grid. Rules determine the birth, death, and survival of a single creature according to the number of beings in adjacent cells. In the Game of Life each cell is either dead or alive and changes its state depending on the states of the cells in its immediate neighbourhood, including its own. The rules which direct the game of life determine that:

- A cell that is alive in one step, will stay alive in the next step when two or three cells among its eight neighbours are alive.
- If more than three neighbours are alive, the cell will die.
- If fewer than two neighbours are alive, the cell will die from loneliness.
- A dead cell will resurrect when surrounded by three live neighbours.

Significant research into employing cellular automata in the generation of musical structures began in the late 1980s (Ariza, 2007). Composers such as Iannis Xenakis, attracted by the simplicity of automata utilised the technique in his piece *Horos* (1986). With regard to synthesis, an approach was taken by Jacques Chareyon which demonstrated the use of one-dimensional automata in the production of waveforms. In his system, the waveform is self-modifying. Each previous waveform becomes a cellular lattice, it is processed through transition rules to create the next waveform (Chareyon, 1990; Ariza, 2007). The system outlined here is primarily focused upon the application of cellular automata to granular synthesis techniques. This was previously covered by Eduardo Miranda with his system *ChaOs*, and earlier by Peter Bowcott (Miranda, 1995; Peter Bowcott, 1989). A detailed overview of historical musical and technical applications of cellular automata was completed in Dave Burraston and Ernest Edmonds (2005).

Now the background to automata has been covered, how the application covered here was approached can be considered. First, the data from the automaton had to be generated. To do this, the Jitter object `jit.conway` was utilised (figure

2). This allowed for a simple model with which to generate the data to operate the sampling system. `jit.conway` was an ideal way to generate the data for the system due to the simple way it can be realised, and that multiple rulesets may be employed and changed dynamically. In a similar fashion to the *Stochastic Sampler*, this application also utilises the `poly~` object and the Java API to increase efficiency.[28] As the player system discussed previously was based largely around granular synthesis, this was the medium for sound output.

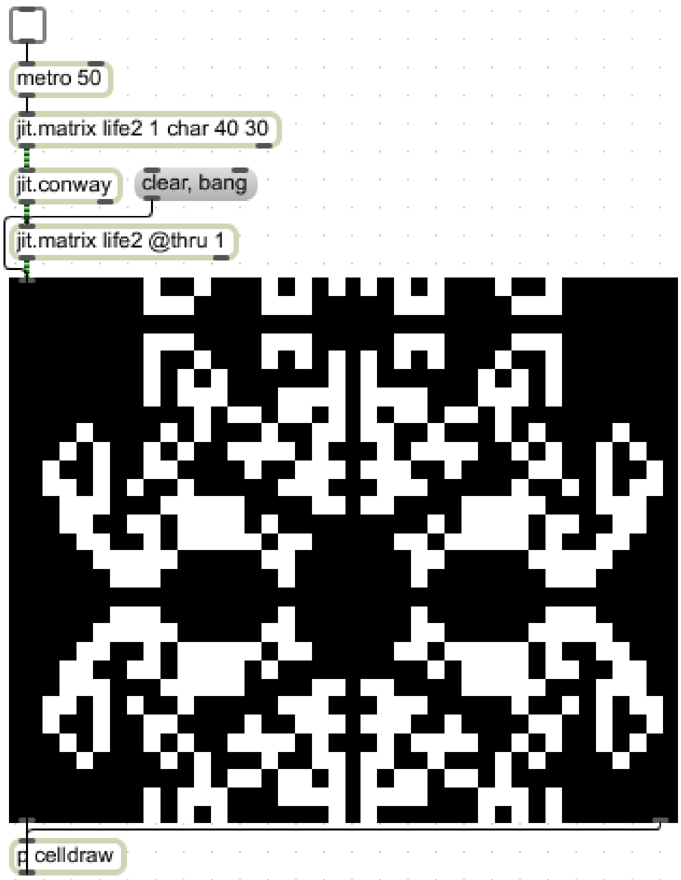


Figure 2: Example of cellular automaton in MaxMSP/Jitter

The cellular automaton was mapped to the playback module by averaging the values of rows and columns. As each cell has a possible value of 1 or 0, this means the system captures the density of alive cells. Conceptually, the mapping strategy was relatively simple. Column density is mapped to the amplitude of a single voice, row density is mapped to the pitch. This allows for the full automaton to

[28] As with the *Stochastic Sampler*, a SuperCollider class was written which improves upon this earlier realisation.

be translated simply into terms that can be applied to the transposition of input audio signals. However, Peter Bowcott, in his 1989 article "Cellular Automation as a Means of High Level Compositional Control of Granular Synthesis" suggests that a mapping strategy as simple as that applied in this circumstance may not fully reflect that the data was obtained from a cellular automaton (Peter Bowcott, 1989). Another approach to the mapping of data from the automaton to a process of granular synthesis was made by Eduardo Miranda in his system *ChaOs*. In this system the automaton was divided into sections. Each section in *ChaOs* represents an oscillator, this oscillator produces sine waves whose frequencies are determined by the mean of the values of the cells in the section. The density of life within these equal sections determined the output of his system (Miranda, 1995). However, both of these approaches were concentrated largely upon the pure synthesis of sounds using oscillators. My approach differs radically in that live audio material is used as the source, thus allowing for a simpler approach to yield relatively complex audio output. This is compounded by the sampling method which is at its heart stochastic, therefore if the automaton repeats the same thing twice there is a probability the sampler will playback or sample the input audio. With regard to Bowcott's comment that a simple mapping strategy does not clearly represent the system, one must consider that samplerate transposition of input audio is a very noticeable effect. This means that the overall density of the system has a audio representation in both pitch and time through the change in playback speed. When this density shifts through life and death of cells in the system, the output sound clearly represents this.

3.5.3 *Swarm Spatialisation* (2009)

Another algorithm utilised in the creation of the *Stochastic Sampler* which has not yet been covered is that of swarming. This particular application was fully integrated with the Markov chain based MaxMSP system; this integration will be discussed. However, it was designed with the idea that it could be useful for different systems in future, therefore it can be considered as a separate development process.[29] Swarming systems are dynamic networks of many interacting agents. Some examples of this are ecosystems, financial markets, cities, or that which can be found in nature, such as birds, or insects (Volker Grimm *et al.*, 2005). The most famous example of a synthesis of this system is Craig W. Reynolds' *Boids* system.[30] His main interest was in a believable simulation of bird flocking. Using three simple rules he determined steering behaviour for each entity, or *boid*. Reynolds' rules were based upon collision avoidance, velocity matching, and flock centring. Depending on these parameters a single entity would attempt to avoid collision with those nearby, try to match their velocity, and attempt to stay close to them (Lebar Bajec *et al.*, 2007). The way swarming systems were applied in this context was as a spatialisation system.

Spatialisation has been a topic of study for some composers working with electronic means, and many different techniques have been developed for its realisation (Malham, 2001). In some cases complex spatialisation techniques have been applied to purely acoustic music as well, such as that of Henry Brant (Harley, 1997). Spatialisation was applied to the *Stochastic Sampler* system in order to translate its dynamic musical approach, to one of algorithmic spatial sound diffusion. To achieve this, Reynolds' *Boids* system was applied to a system of spatialisation using MaxMSP and the `boids3d` object. An Ambisonic (Malham and Myatt, 1995) method was used to spatialise the *boids*, utilised with the help of the ICST ambisonics externals[31] for MaxMSP. Ambisonics was chosen as the method for spatialisation of this system due to the precise method in which points can be represented using cartesian coordinates. In addition to this, the system may be scaled in order to utilise larger or smaller speaker arrays with ease. A mapping system was developed in which the coordinates for each *boid* would be mapped one-to-one to a point in space through the ambisonics system. Parameters for the *boids* are controlled through a MaxMSP patch, and a visual representation

[29] In addition to this, a port of the "BoidRoids" swarming simulation bundled with MaxMSP was developed as a SuperCollider extension. This allows for application of swarming algorithms within the current framework. The code for this implementation can be found online at: `http://sccode.org/1-4RY/` accessed: 21/12/2012

[30] `http://www.red3d.com/cwr/` accessed: 01/03/12

[31] `http://www.icst.net/research/projects/ambisonics-tools/` accessed: 20/01/2012

of the sound output is given.

This spatialisation tool was added as a module for the *Stochastic Sampler*, and certain parameters were linked to its output. Each voice of the system was given its own *boid*, and therefore point in (x, y, z) space. The concept was to further the system, allowing for it not to operate dynamically in sound, but also in space. In order to integrate the swarming algorithm to the sound in a coherent manner, some basic statistical operations were done on the musical parameters to map them to the *boids* arguments. Mapping was relatively complex in this system, this was to try and create one entity rather than a system with spatialisation tacked-on.

Average deviation was taken from all Markov pitches, this was scaled linearly into usable ranges for *boid* parameters: willingness to change speed and direction; preferable distance from neighbours; number of neighbours consulted when flocking; strength of speed matching instinct; strength of centring instinct; strength of wall avoidance instinct. Average deviation was used in order to translate the concept of a single swarming entity to the pitch system; the more deviation from the mean in the pitch system, the more likely the swarm is to act irregularly. This was compounded by using the mean of the Markov pitches to determine the point to which the *boids* were attracted to, and the strength of their neighbour avoidance instinct. Rhythm was also utilised as a mapping parameter, this was done in such a fashion that the overall density of attacks determine the overall speed of the swarm. Statistical skew of the rhythm triggers determine the acceleration of the system. This has a relatively simple application, the quicker the output, the quicker the swarm moves. However this can be controlled by the user with a parameter, this determines how many past attacks are taken into account by the spatialisation system. In essence this slows the response of the swarm, reducing the chance of the system moving too quickly.

Location of the individual *boids* can be relatively difficult depending on the source sound, number of speakers, speaker placement, ambisonic-order, and transposition. Many variables affect the *resolution* of the spatialisation, however movement is present in every case. As the system was designed to truly bring the swarming system together with the musical system, location of individual voices was overshadowed by a concentration on recognition of the swarming behaviour itself. This was to give the grain cloud a spatial dimension, allowing it to behave like an organic being in space.

4 Realtime Systems

In this section the systems written for the realisation of the generative pieces in the composition portfolio will be considered. Here, the pieces will be approached in a relatively technical fashion with regard to the systems and signal processing chains that create them. In the next section the pieces will be looked at in a more qualitative fashion, with regard to the composition of the pattern systems. These systems mark a clear development from the earlier work done on the *Stochastic Sampler*. Ultimately, this section provides an overview of generative composition technique, and the work completed towards a standalone generative production and performance tool. Real time systems for generative music have arisen in the relatively recent past, mainly due to the rapid increase in processing power available to composers. Robert Rowe, one of the most well known creators of interactive music systems describes such systems as:

> *[...] applied music theory; ideas about the description and generation of music are formalised to the point that they can be implemented in a computer program and tested in real time.* (Rowe, 1992, pg. 43)

The term "generative" in the context of composition can be defined as a system which happens to produce an output in real time (Collins, 2008; Brian Eno, 1996). This is in contrast to algorithmic composition wherein one may operate on a non-realtime basis. Broadly, in generative art, the artwork is generated by a process that is not under the direct control of the artist (Margaret A. Boden and Ernest A. Edmonds, 2009). This process may be a part of a whole, or a larger element. In computer music a generative model allows for the composer to apply a general method in the form of an algorithm to produce a structured set of compositional parameters. However any rule-based system such as Stockhausen's *Aus den sieben Tagen* may be labeled generative music by some (Nick Collins and Andrew R. Brown, 2009). Generative music is not a new idea and there are releases in the mainstream such as Autechre's *Confield* (2001). What is new about the approach considered here is the overall methodology for production, performance, and delivery of this media. This is considered throughout this chapter.

4.1 The Composition Methodology

Continuing from the *Stochastic Sampler* in a relatively technical light, the system developed and the resulting composition methodology will be looked at. It can be debated whether this should be within a section on "real time systems" as the structure itself is static. However, the output is generative and this is the focus of the discussion. This 'framework' represents the basis for the compositional methodology utilised for the generative pieces. The pieces discussed later in this section provide the background to the development of this system. The system has been continuously updated, however at present it operates extremely reliably and provides a useful framework for composition. The system in its present form was used to compose *Stratus*, prior revisions were used for *pp*, and *For Putten*. Overall the concept is extremely simple, separate functions are bundled together into a number of files which group them into categories. These files are outlined below:

- `user_interface.scd` A simple user interface operable by the local user
- `osc.scd` Remote Open Sound Control functionality (see section 4.2.2)
- `startup.scd` Loads all necessary files, initialises system for playback
- `synths.scd` Contains all signal processing and synthesiser code
- `audio.scd` Routing, buffers, and persistent synthesiser information
- `patterns.scd` The score, all sections are contained in this file[32]
- `sequencer.scd` Macrostructure, functions for playback and stop
- `mixer.scd` Channel strip, and output
- `ct_event.scd` Event type for conTimbre OSC interaction

[32] Occasionally this will have to be enumerated (patterns_1, patterns_2...) due to a limit on function size in SuperCollider.

- `ct_orchestra.cePlayerOrc` conTimbre orchestra file
- `ct_orchestra_loader.maxpat` Automatic orchestra loader

Interaction with the system by the end user is done through the user interface. This was originally implemented in the system in its revision for the piece *For Putten*. This interface allows for the user to interact with the system in a relatively simple fashion, rather than having to execute blocks of code by hand. In addition, this interface is able to load the other files in the correct order. This includes initialising MaxMSP and loading the correct orchestra into conTimbre, starting the jack audio daemon, and routing all audio channels between MaxMSP and SuperCollider[33]. The interface itself is shown in Figure 3. It has been designed not to look particularly exciting, but very simple. The user needs only to press "init", and the system will be ready for playback. The buttons for conTimbre initialisation and audio routing are there for troubleshooting and testing purposes. A lack of volume control may be noted, however this is provided on the window for the SuperCollider server itself by default.[34] Pertinent messages regarding the transition of a musical work through sections, and error messages, are shown in the SuperCollider post window rather than on the user interface.

Figure 3: Simple user interface

For orchestral pieces utilising conTimbre, the system interfaces with conTimbre using the Open Sound Control (OSC) protocol. While MIDI was originally utilised as the primary control protocol in pieces before *Stratus*, it has now been

[33] This has only been tested on Mac OSX, however should be compatible with Linux systems
[34] SuperCollider v3.5

superseded. The reason for this change was for increased functionality within conTimbre, and the ability to send floating point note values. This means it is possible to compose using complex scordatura with no problems. As there was not originally any support for OSC functionality through the event framework, an event type was written for SuperCollider. This handles the OSC commands being sent to conTimbre directly from a pattern. Each of the particular note types for conTimbre are supported, along with automatic handling of note off messages. This provides an extremely simple way to operate the sampler through the event framework.

Musical composition using the framework is not any different from using patterns in SuperCollider in any other way. However there are some unique elements due to the proprietary event type. A simple example of a standard instrumental section as written in the software is shown below in Figure 4. This is a simple pattern and represents the viola part in the first section of *Stratus*. Here, the event type `ctosc` has been chosen. This proprietary type sends OSC messages to the value of the key `oscout`, in this case that is the global variable `~oscdestination` which corresponds to a `NetAddr` pointing to the sampler. In this system, the key `voicename` corresponds to an instrument on the conTimbre sampler which will respond to that particular name. In this case it is `va1`, or viola 1. To this instrument, the event is sending commands for rest (handled by the event type itself) and note-on (sent to the sampler). These commands in this case are handled by a value stream which will first output a rest, then either a rest or a note-on at random. The other parts of the event are concerned with musical output and are discussed in the later section on patterns.

```
// viola
    ~delays[0]+0.05,
    Pbind(
        \type, \ctosc,
        \oscout, ~osc_destination,
        \osccmd, Pseq([\rest,Prand([\rest,\noteon],inf)],1),
        \voicename, \va1,
        \midinote, Prand(~hseries[0].select({|n,i| n>=81}).select({|n,i| n<=96}),inf),
        \dur, Pseq([Pn(16,1),Prand([8,16,32],inf)],1),
        \amp, Pexprand(0.1,0.7,inf)
    )
```

Figure 4: A simple instrumental section

The ∼`delays` array in figure 4 provides an individual per-instrument time offset for each section using the `Ptpar` pattern system. This is important because it embeds each of the event streams together, allowing multiple streams to be played together. This is the basis of the polyphony used in my compositional approach, and can be seen in Figure 5.

In Figure 5 another command handled by the `ctosc` event type is shown. These are the program changes which determine the timbre of each instrumental voice in the sampler. Program changes are not delayed by the section time offset. This is because if two messages are sent to the server at the same time, they may execute in any order. By offsetting the instrumental parts by 0.05 it is possible to be sure that the program changes will be sent to the sampler first.

```
// program changes
    ~delays[0],
    Pbind(
        \type, \ctosc,
        \oscout, ~osc_destination,
        \osccmd, \program,
        \voicename, [\fl,\bfl,\bcl,\sxl,\val],
        \programname,
            #["flute.air noise.closed.vowel varied",
            "bass.flute.multiphonic",
            "bass clarinet boehm system.ordinario",
            "alto saxophone.slap.percussive slap",
            "viola.harmonic.artificial.fourth"],
        \dur, Pn(0.01,1)
    ),
```

Figure 5: The wrapper for a section, and the initial program changes

A further element shown in Figure 5 is how each section in the system is controlled in time. Each section of the piece is an element of the array ∼`sections`. This array element contains three basic levels of structural hierarchy: `Pfindur`, `Ptpar`, and `Pbind`. The top of the hierarchy: `Pfindur` determines the length of the section, it will stop the event streams within when it reaches the time contained in the corresponding element of the ∼`durations` array. The second level of the hierarchy: `Ptpar` plays a number of `Pbind` patterns in parallel according to the time offset given by ∼`delays`. The third level of the hierarchy: `Pbind` contains the section specific patterns for each instrumental voice and determines what can be considered to be the microstructure of the piece. Each of the elements contained in

~`sections` are easily accessed by calling ~`sections[n]`. As this macrostructure operates on a completely different level of the hierarchy, it becomes extremely simple to develop complex forms and generative structures.

Audio routing in the system went through a number of revisions, the major being that required for *Stratus*. As there were many more instruments than earlier pieces, the number of audio tracks required for routing increased. Therefore routing in this system was achieved using Jack rather than SoundFlower. This setup allowed for 16 channels of dry instruments and 16 channels of signal processing. The routing was approached in a similar fashion to that of a live-electronics concert, in this context each track represents what would be a microphone in a live situation. Each of these dry audio streams are sent to an audio bus, which is used to routing the audio to the signal processing where necessary. Signal processing in the piece is achieved in a similar way to that of instrumental sections, and is bundled in a similar way with the instrumental patterns. An example of a simple signal processing element is shown in figure 6.

```
// flute -> warp
    ~delays[0]+0.05,
    Pbind(
        \instrument, \warp,
        \group, ~fx,
        \in, ~master_dry_bus.subBus(0,1),
        \out, ~master_fx_bus.subBus(0,1),
        \dur, ~durations[0],
        \atk, ~durations[0] * 0.1,
        \sus, ~durations[0] * 0.3,
        \rel, ~durations[0] * 0.8, // slight overlap with s2
        \amp, 0.65,
        \warpfactor, (3,5..11).midiratio,
        \freqscale, Pkey(\warpfactor)
    ),
```

Figure 6: A signal processing pattern

The main difference in this pattern is the event type, which has been substituted for an instrument. This instrument corresponds to a synthesiser, these are contained in the `synths.scd` file. In this case, the synthesiser is the warp effect which has been utilised quite frequently. It is possible to expand signal processing patterns using arrays, as can be seen with in the `warpfactor` key[35], this highlights the concise nature of this method. The `group` key corresponds to where

[35] Here (3,5..11) expands to the array [3,5,7,9,11]

the synthesiser will reside in the signal chain on the server. In this framework there are four groups, ~`input`, ~`fx`, ~`hala`, and ~`output`. For most purposes ~`input`, ~`fx`, and ~`hala` are accessed much more regularly than ~`output` which is only used for the mixing procedure and channel strip synthesisers.

Output from the system is handled using `mixer.scd`. Output from all signal processing synthesisers, and all dry instruments are sent through an audio bus to their respective output synthesisers residing in the ~`output` group. The output synthesisers operate in a similar way to a simple channel strip. Compression, panning, equalisation, and amplitude parameters are given to each voice and are set by hand in the `mixer.scd` file.

4.2 Challenges

Now the methodology of the approach has been covered, the pieces created with the system can be considered. However, before this, the perceived challenges of real time music production and how these have been tackled will be looked at. Real time systems have some major advantages for electronic music performance. However, there can be some limitations specific to this field, Risset (1999) states these as:

- Realtime systems have a limited level of complexity. The limit of complexity leads to an insufficient timbre quality in comparison to that of a large orchestra. Because of this limit the composer does not have the freedom to choose the level of complexity that may be desired.

- Realtime synthesis systems are more inflexible than software synthesis.

- Realtime systems are not the solution to the problem of mastering digital synthesis. Risset states that "[an inexperienced composer] believes that real-time operation will enable him to tune the result so as to achieve the desired musical sound, using his ear and his intuition"

- Realtime systems deprecate quickly. New instruments have a short life, even though they are musically useful. This brings a risk of perishable, memoryless electronic art.

Risset states when writing these limitations that he was playing the devil's advocate, and it is clear they were perhaps more applicable at the time of initial writing. Nevertheless, it is helpful to understand the challenges a computer musician may face when designing a system for composition. These limitations give the programmer of interactive systems some specific challenges, and some interesting questions. In part, these bullet points give the designer of computer music software some specific concepts to focus on when developing their systems.

Regarding system complexity, history has shown this is a function of time. As technology progresses, the headroom for complexity rises. Coupled with efficient, well designed systems, the possibility for more complex compositional or signal processing constructs increases. This places some focus upon the design of the overall system itself: how efficient it is, how well it is dealing with the tasks given to it. Therefore when designing a system for a specific task the composer/programmer must consider the overall system and how it relates to the capacity available. With relation to the *Stochastic Sampler*, this issue was tackled by porting the complex workload to Java. The concept was tackled in pieces such as *pp* by effectively managing groups of signal processing chains, sending them to the server only in the relevant sections.

Given the challenge of deprecation, this is also a concept which is dependent on the complexity and musical validity of a system. It is not dependent on whether the system is real, or non-realtime. For example, when frequency modulation was conceived by John Chowning in 1973 it was a non-realtime process (John Chowning, 1973). Even now frequency modulation has clear uses in synthesis. Pieces such as Chowning's *Stria* (1977) have not aged particularly badly, and are still within the sphere of discussion after 30 years (Meneghini, 2007; Zattra, 2007). However, these pieces and the techniques utilised by synthesisers of the 70s would be easily replicated in real time systems today. It is clear that here the challenge is in developing an interesting and practical musical instrument, rather than concentrating on the technique in which it is created.

Another challenge touched upon by Risset is that the composer of real time music may not achieve the sound he desires through intuition. This can be extrapolated to some degree to include the concept of mixing and mastering of real time music which will be considered more thoroughly in the next section. More directly this could apply to specific methods of mixing, mastering, or specifically digital synthesis. Clearly through time this method has been utilised repeatedly, standard equalisation practice is to tweak in real time in order to fix problem frequencies (Bob Katz, 2003). This could be as important to achieving the correct sound from a digital synthesiser as the modification of frequency modulation or other, more direct systems. By experimenting in this way it is possible to achieve sounds that were never considered previously, and concepts can be discovered through accidental modification that inspire further creativity.

4.2.1 Mixing and Mastering

One of the issues mentioned by Risset regarded the mixing and mastering of real time systems, this is an important issue which will now be considered. A complication with the release of generative music is achieving the same quality of sonic material as would be expected from its recorded counterpart. As one can never be completely sure of the actual output, it can be extremely difficult to master and mix effectively. This is largely related to the level of constraint applied to the processes governing signal processing and dynamics applied to the piece. There are some ways to tackle this issue of quality, but no absolute solutions that I have currently found. With the correct application of dynamic processing and mastering techniques output may approach the quality level of a recorded master but likely will not match it.

On the most fundamental level, the easiest way to achieve mixing cohesion is to split the tracks into their separate elements. By splitting up the tracks in this fashion, problem frequencies or difficult signal processing techniques may be tamed using standard mixing practices. For example, the convolution of violoncello and square wave in *Stratus* presented some specific issues. These issues were apparent when the square wave played frequencies close to the fundamental resonance of the cello, and lead to some difficulty with controlling the output level. Here, the solution was to reduce the level of the output convolution at the problem frequency by around 3dB with a bandwith of around 1.5 octaves using a parametric equaliser. This fix could then be applied to the output stage of the piece within the ∼`output` group, or hard-coded into the convolution synthesiser itself.

Continuing in this vein, one may even consider metering techniques and proper loudspeaker calibration as specific elements which can be of major help when working with generative music. From the perspective of a experimental or classical piece, my personal preferred metering technique is that of Bob Katz's K-20 system (Bob Katz, 2003). Here 0 dB is tuned to -20 dBFS where 0dB represents approximately 83 dB SPL. This gives a level of 103 dB SPL at 0 dBFS, which is loud. An average mixing level of 0 dB is aimed for, or 83dB SPL which represents the most linear response from the human ear according to the equal-loudness curve (Harvey Fletcher and Munson, 1933). So using the K-20 system a relatively large amount of headroom if left, minimising the requirement for limiting and compression to allow for a much more natural response. As my personal approach to mixing leads to a relatively large crest factor, this metering method is perfect. Of course some slight compression may be necessary to bring this crest factor to levels which are suitable for personal listening. Here very slight compression to the mix bus may be applied, and some overall equalisation for sheen.

However, in the studio there may be expensive outboard compressors and equalisers that cannot be printed onto the output sound if it is generated remotely. This provides a big challenge for remote generation of music in realtime. To combat this the overall mix and mastering approach must be hard coded at the output stage. This is an issue which is considered further in the following section on delivery (4.2.2). Overall, there seem to be no clear solutions to the issue of mixing and mastering generative music in realtime with regard to hardware constraints. However, coupling standard mixing practice with a high level of headroom and transparent application of limiting and compression can be beneficial to a system's output if hard coded into the output stage.

4.2.2 Delivery

As touched upon previously in section 2.3.2, the concept of social context when considering music of a generative nature was important to me throughout the research period. This focus upon delivery was developed from an understanding that music requires an audience, and a context within society. Therefore, a delivery method had to be utilised or designed which allowed for music of a generative nature to thrive. This method had to retain all of the advantages generative music provides, such as a contingent nature and focus upon performance-derived qualities, while also being feasible to access for a normal listener. A number of methods were trialed and developed. These experiments resulted in a number of questions and proposals for further research. This section will cover these methods and give an overview of the research process.

Perhaps the major challenge with generative music is how to deliver such a media to the target audience, while retaining a quality comparable to a static CD release. In the case of the music developed for this portfolio, the target audience has been the personal listener. There are a number of possible solutions for delivery we will now consider. The most obvious choice is to package the whole system onto physical media such as a USB stick or DVD-ROM and have the system run locally. However there are some challenges with this: mixing and mastering as discussed previously, and the reliance upon samplers. The problem with samplers is the large size of the sample library required to generate the pieces. This would provide challenges for distribution, especially if proprietary sample libraries are utilised which cannot be split into separate parts. In addition to this, varying system architecture and operating systems may lead to compatibility issues with the playback framework. Regarding operating systems, at present SuperCollider operates on MacOS, Linux, and Windows. However, with varying hardware configurations, one may find that the system does not work effectively (i.e., audio dropouts, or other artefacts not present during testing). In addition to this, it may lead to a less coherent user experience. For example, the user may need to actively change certain variables such as ~`path` to make the playback function correctly.

Nevertheless, it is possible that some of these issues may be bypassed by changing the physical location of certain elements of the system. The systems discussed have been designed in a modular fashion; control systems sending messages over specific protocols to samplers, with audio sent back and processed. This design strategy allows for the sampler to operate from a number of different locations:

- *Locally* Control messages are sent using MIDI over an Inter Application Control Bus (MacOS),[36] or OSC, using UDP or TCP protocols. Audio messages in this circumstance are routed using SoundFlower,[37] or Jack.[38]

- *Remotely (on-site)* Control messages are sent using MIDI using hardware to a separate machine, or using OSC (preferred), using the UDP or TCP protocol. Audio is routed directly using a cable from one machine to the other.

- *Remotely (off-site)* Control messages are sent using OSC over TCP to the remote machine, audio is sent using JackTrip,[39] MaxMSP/Jitter, or similar audio-over-ethernet software.

However, operating the sampler at an off-site location may introduce more problems than are solved. For example, latency issues for audio processing would become extremely difficult to manage. Therefore one solution would be to operate the whole system remotely, a standard user interface much like that of Spotify[40] or iTunes[41] would allow for the user to browse generative media. This would then be processed remotely and sent to the listener over a network using a system such

[36] On Windows the MIDI tools from `http://www.midiox.com/` accessed: 30/07/12 have been tested

[37] `http://cycling74.com/soundflower-landing-page/` accessed: 30/07/12

[38] `http://jackaudio.org/` accessed: 30/07/12

[39] `https://ccrma.stanford.edu/groups/soundwire/software/jacktrip/` accessed: 30/07/12

[40] `http://www.spotify.com/` accessed: 30/07/12

[41] `http://www.apple.com/itunes/` accessed: 30/07/12

MaxMSP/Jitter. A proof-of-concept system was designed in MaxMSP/Jitter to show this concept at work, and is shown in figures 7, 8, and 9.

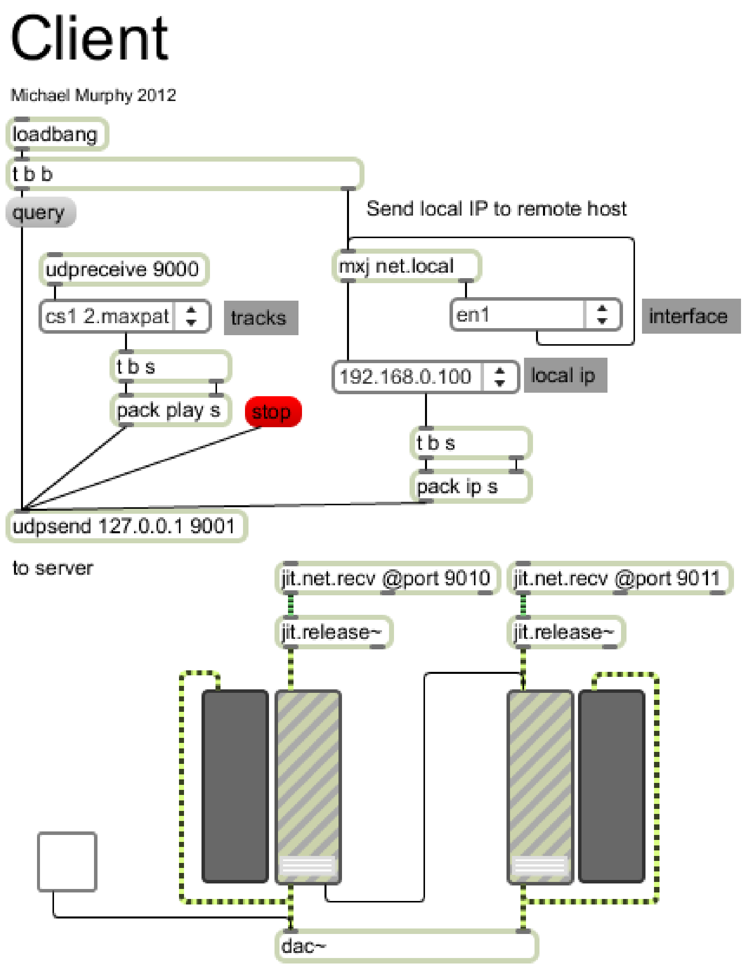


Figure 7: Client in patching mode

This system is based upon a simple client-server architecture. On startup, the client connects to the server, passes it the local IP address, and requests the updated composition portfolio. This portfolio is then downloaded and automatically populates a menu. When the user selects one of the pieces from the menu, a message is sent to the server to execute the system. The server then sends a message to a SuperCollider OSC responder which executes the given command. Audio is then routed from SuperCollider back through to MaxMSP and sent back to the client for listening. While the system seems relatively complex, the concept is quite easily implemented. Essentially this system is an extension of the user interface designed for *For Putten*. Another possibility would be to run a simi-

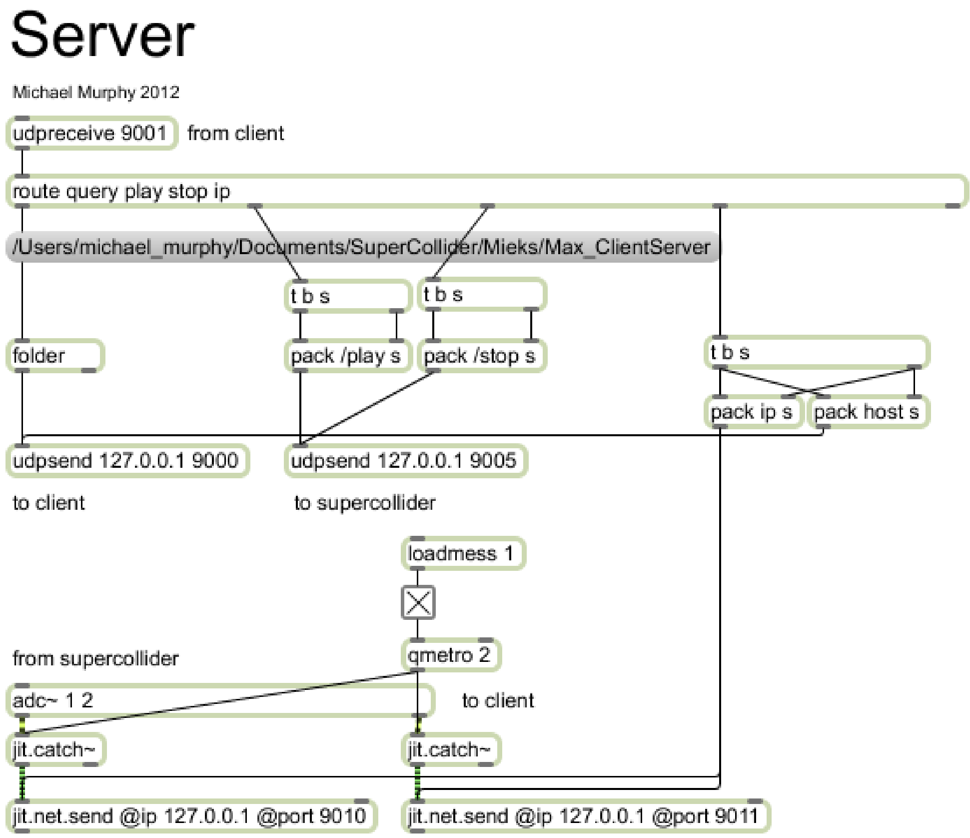


Figure 8: Remote server

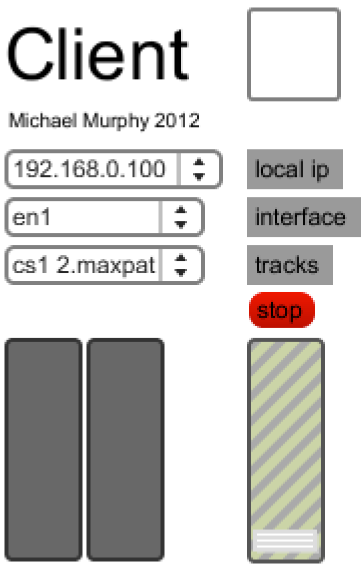


Figure 9: Example of client in user mode

lar server system on a website, this would allow the user to access the material without requiring the client software.

The advantage of using a system of this sort is that there would be no compatibility problems on the users end. The client software is extremely simple and would operate on both OSX and Windows versions of MaxMSP/Jitter, with few third party requirements[42]. In addition to this, as no complex processing is completed on the users side, one may operate the system from extremely low power devices. Future revisions could move away from the MaxMSP environment and allow for a proprietary system to be made in Java using the Java Sound API. This would allow for complete platform independence, and also provide the opportunity for expansion onto mobile platforms, primarily Android based devices.

However, there are some inherent issues with a system of this sort. Primarily, the issue is that as the user does not have access to the system itself, it is therefore unusable without an internet connection. This presents a paradigm shift in how contemporary music is consumed: a service based system rather than owning a physical copy. Already this trend is apparent through the growing popularity of sites such as iPlayer, 4oD, Last.fm, Grooveshark, Spotify, and Pandora. These services provide a large media library to their subscribers for a price[43]. Internet radio is the perfect example as the user is never given an option to download a digital copy. Keeping music on the "cloud" allows the user to access it at any point, solving some problems with generative media, specifically that one must be at a computer to generate the music. A further issue with a system such as this is scalability. The server system would need to be designed to support user concurrency, automatically spawning SuperCollider instances to provide an individual reproduction for each user's client. Furthermore, network bandwidth would quickly become a major concern. There are clearly some complex technical challenges inherent in this idea, however there are some other approaches to this problem.

The primary alternative approach takes the concept of decentralising the sys-

[42] Java must be installed for the `mxj net.local` object.

[43] In the case of the BBC this is the TV licence; other services utilise advertising or subscription to generate revenue.

tem a step further. A similar concept has been approached before, for example Jem Finer's *Longplayer* which transmits over the internet[44]. In this case the stream is unbroken much like a radio station; a standard audio player such as Quicktime, VLC, or Foobar 2000 provides the client functionality. Indeed, at the time of writing if one types "generative radio" into a popular search engine many possible options will appear. This approach alleviates many of the problems inherent with a standard delivery approach, with a trade-off in that user control is almost completely removed. Another advantage is that this approach would allow for fully mastered and mixed pieces to be played over tested hardware to the audience. A standard implementation of this idea would operate the whole system using a playlist, and transmit the output using a system such as Icecast.[45] The audience would tune in using a web browser or compatible software and listen in a similar fashion to any standard radio station. To demonstrate this concept, a proof-of-concept system was developed using four software packages, Icecast, Darkice[46], Jack, and SuperCollider. In this system, SuperCollider provides the source audio. The audio playlist is configured in a similar way to the OSC system, where a meta-process sends OSC commands to play pieces written using the composition framework. This source audio is then routed through Jack to Darkice, which interfaces with Icecast. Icecast in this system is the radio server, which allows for a remote user to connect to an MP3 encoded stream on the server at the specified port. To listen to the stream, any compatible client may connect and it will be broadcast at the specified bitrate.

Regarding the actual content of such a radio station, what appears to be the standard generative approach to radio delivery tends toward infinite length pieces. Examples of these are *Longplayer* and *Patchwerk*[47]. In the case of *Patchwerk*, ten-minute sections are played which crossfade between each other randomly never allowing for a break, in *Longplayer* a similarly never-ending approach is utilised. This never ending approach to generative music seems to have become almost

[44] `http://longplayer.org/listen/longplayer.m3u/` accessed: 01/07/12
[45] `http://www.icecast.org/` accessed: 09/06/2012
[46] `http://code.google.com/p/darkice/` accessed: 07/09/12
[47] `http://patchwerk.rumblesan.com/` accessed:10/07/2012

a standard paradigm, and therefore a more traditional application of generative systems to music may provide contrast. To bring the "concert" generative piece back to the centre stage, a large amount of creative material would be required. Therefore one proposal would be a website wherein the user would upload their software in a similar fashion to the highly popular SoundCloud[48]. From there the piece would be moderated, tested using a virtual machine to prevent malicious application, and entered into a probability based polling playlist system for playback. This concept would not only allow for a clear social context for generative music, but also provide a stage for this music with an audience.

4.3 Generative Music: The Pieces

Now a background has been covered along with many of the more technical aspects, the pieces developed using the system described in section 4.1[49] can be examined. This portfolio of composition represents the body of work which lead to the development of the current version of the composition system. Here the pieces will be considered primarily from a signal processing and technical viewpoint, looking at how the process of composition lead to the further development of the overall composition methodology.

4.3.1 *pp* (2011)

pp was written in 2011 for automatic prepared piano. The composition was realised using a combination of the Kontakt sampler software, and the SuperCollider environment. *pp* was designed to be released as standalone software, it was the first result of the work completed towards a dynamic composition release. It represents a system developed in a realtime context, designed to be listened to in a realtime context. *pp* was written to be listened to live, and each execution of the system should be slightly different in the ways defined by the algorithmic constraints. The composition itself was based upon a unified system of signal

[48] `http://www.soundcloud.com/` accessed: 01/07/2012
[49] Variations of the system; the composition framework was developed using the composition process as feedback throughout.

processing and acoustic note production. Technically, these are fundamentally entwined through the control software. In *pp* this control system was written with a dual purpose, to allow for possible concert performance as well as personal listening. To do this it was designed in a relatively simple fashion. Simply, the system sends MIDI notes to the instrument it is controlling and receives audio; the control system controls all note production, signal processing, and audio output.

When designing the system a decision had to be made as to which signal processing parameters would be modulatable. Also, who was to control this modulation. For instance, with regard to the piece moving through sections: a foot pedal could be utilised and given to a performer, or it could be left to a sound projectionist, who would be reading from a score in tandem with the performance. However, as the concept of musical automata and generative music for the personal listener was the primary goal, it was decided this should be completely automated. This also had the compositional advantage that the structure of the piece could become dynamic in specifically formalised ways that would be difficult otherwise. This is discussed in section 5.4.3, which concentrates on the non-realtime elements of the piece.

Signal processing in *pp* was based upon a serial design rather than a parallel one. Therefore the output from each element in a chain fed into the following element. This was applied to suit the sound desired from the composition itself, wherein only one serial signal processing chain would be applied for each section. Sound design was based upon this principle of layered processing, interacting with the acoustic sound on a per-section basis. This serial design did not hinder the processing or production of the recorded sound, mainly due to the way the sound design was implemented. As the synthesiser definitions were usually written individually for each section, most of the signal processing could be kept within a one or two elements of the chain. However, some regularly used elements such as panning, pitch shift and delay were re-used throughout.

All output from the system ran through a synthesiser for panning and spatialisation. A sine oscillator was used to control the position of these sound sources in two dimensional space. The sine oscillator was chosen as the primary method for panning as the calculations necessary to produce multiple types of movement and panning are relatively simple. To achieve the panning itself, the `PanAz` object was used. This object spaces channels evenly over a cyclic period of 2.0. Therefore, all channels will be cyclically panned through if the panning position is modulated using a sawtooth oscillator with the range -1 to +1. By using a sine oscillator, it was fairly simple to control movement or static position of sound by modifying the frequency and the phase, multiplication, and addition factors.

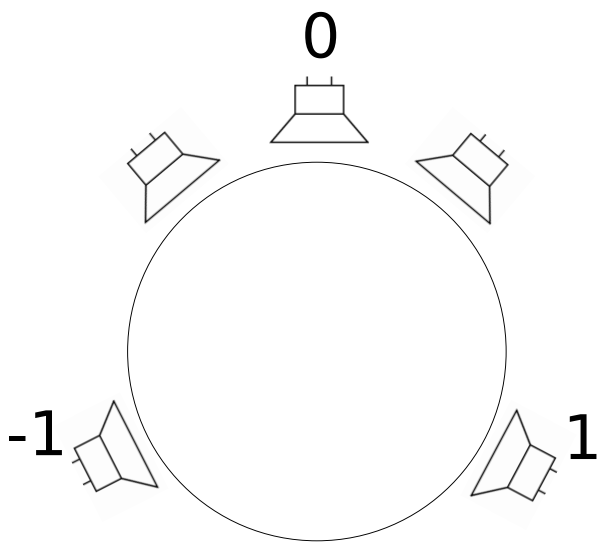


Figure 10: Rough panning example

Now the signal processing elements and audio routing have been covered, the content of the piece must be considered. Much of the piece is built from recorded samples, re-transposed through playback rate modification. This is a running theme throughout the piece and stems from an interest in creating a complex temporal output. The same technique is used in the *Stochastic Sampler* as described in section 3.3.2. Two buffers were all that were necessary to capture the samples required, however they are often played back simultaneously. Therefore a section may be recorded into buffer one and played back in the next section in five separate tempos, while the other is recording the dry piano sound. The idea for this was to retain continuity and musical relevance, while also providing some development through polyrhythm and harmonisation.

In addition to the relatively simple sampling technique, there are some other signal processing ideas present in *pp*. One of the more complex elements was a system derived from cascading delay lines. In essence this system takes a number of delay lines which feed into one another, and outputs the sound. To create a higher probability of interesting overlap, delay period is individual to each delay line. For each level in the delay network the range of possible delay times is compressed. This effect can be heard from ∼8'30 in the piece; periodicity in the output is noticeable from the staccato section, but this shifts to a more reverb-like quality when the piano sound re-emerges.

However, many of the complex textures found in *pp* were based upon serial signal processing chains rather than single elements. For instance, section six in the piece (∼6'30) comprises of three frequency shifters in glissandi, and three samplers playing back a sample from the previous section at different tempos. The tempo of this playback is dependent upon the duration of section six, and the preceding section. Thus, it has a direct correlation with the transposition, which is affected by the change in playback rate. Where d is equal to the previous duration divided by the current duration, the tempo ratio can be expressed as:

$$d : \frac{d}{1.3} : \frac{d}{1.5}$$

In order to retain a musical connection to the original timbre, many of the signal processing effects utilised throughout the piece were based upon pitch shift or delay. This was to provide a link to the source material, which is often relatively sparse and in the background. This sparseness is developed and manipulated in realtime using electronics and recordings to create the final sound.

As *pp* represented the first working iteration of a standalone dynamic composition for this portfolio, what was learnt from this system will be considered. There were certain limitations and nuances which became apparent from this development. The main difficulty when designing *pp* was mixing and mastering[50].

[50] This is covered in depth in section 4.2.1

This was catalysed by the serial nature of the signal processing chains employed. In *pp* a final node was added to the end of the signal chain to implement limiting for a standalone release. This provided a “safety net” to protect the listener’s audio equipment from any elusive audio events which may have occurred. However, many of the concepts touched upon through the development of this system were expanded upon and still are apparent in the final system.

4.3.2 *For Putten* (2012)

Written for sampled percussion and live electronics, *For Putten* represents the continuing evolution of the composition environment designed for *pp*. This piece was the second iteration of a generative music release, and ironed out many of the flaws in the original version. Furthermore it was the first piece to use the conTimbre sampler, allowing for more varied instrumental samples to be utilised. However, instrumentation in this piece was relatively minimal, comprising of wood blocks, conga, bongos, bass drum, kick drum, log drum, and thundersheet. Live electronics took a large role in timbre modification, and there was an added electronic component in the form of a subtractive synthesiser. The synthesiser itself was relatively simple, taking white noise as source material, a resonating filter was applied at a specific frequency to create the output sound. This synthesiser also utilised concepts of frequency and amplitude modulation to generate the output sound.

In a very technical light, the main issues in *pp* were ironed out with this release. This was primarily the serial signal processing chain. Signal processing in *For Putten* was a more traditional parallel setup, with each effect and instrument occupying its own track. Serial signal processing is still possible with the system and appears in the piece in multiple sections, but overall the approach was a parallel setup. This more traditional approach means the system was much easier to control for mastering and mixing purposes. The quality of the output was far more consistent and easier to tackle from an engineering standpoint because of this separation between channels. Like *pp*, the piece was designed to run and be processed completely in-the-box. This was achieved by routing audio between the respective applications: conTimbre, SuperCollider, and Reaper, with routing handled by SoundFlower. This method of output through digital audio workstation was deprecated after the development of the system for *Stratus*, which allowed for all mixing and output to be controlled through SuperCollider.

The signal processing elements of *For Putten* are relatively simple, relying on effects such as reverb, samplerate transposition, frequency shifting, and pitch shifting throughout. Pitch shifting was used within the piece not only to transpose, but also to achieve a granular synthesis style effect. This was done by modification of the pitch and time dispersion parameters available. For example, the introduction to the piece features three filtered pitchshifters, with time dispersion set to one quaver, and pitch dispersion to a random value. The result of this is a granular texture. Conversely, the same pitch shifter was used to transpose the bass drum down by one octave in the following section. The best way to show this is with the output from a spectrogram, in figure 11 the granular activity from the pitch shifter can be seen. Here the beginning of section three shows a granular texture created by utilising eight parallel pitch shifters with a large pitch dispersion range.

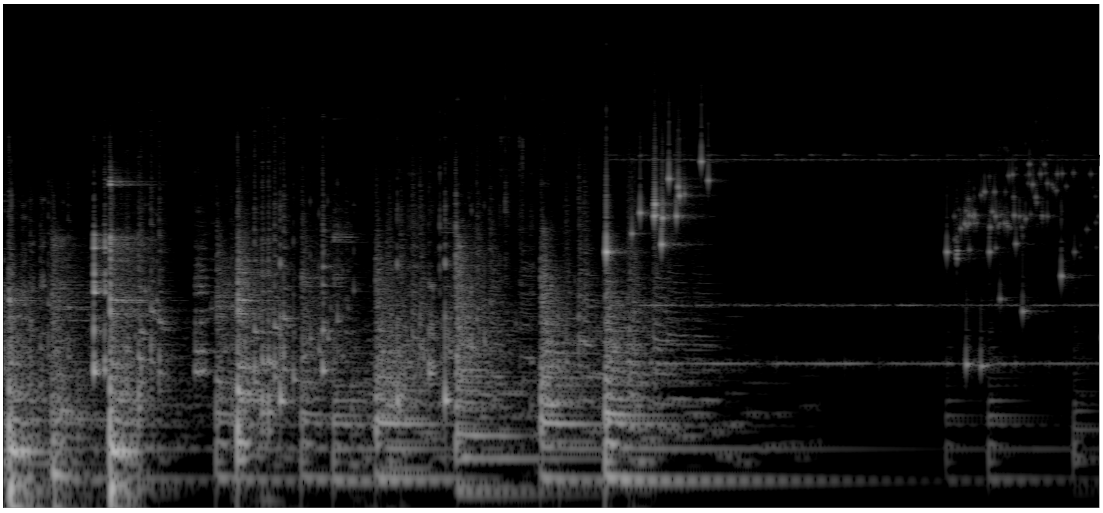


Figure 11: Granular behaviour from pitch shifter in *For Putten*

By far the most used signal processing tool in this piece was the *WarpIn* unit generator. This granular time stretcher and pitch shifter was used to achieve a symbiosis between pitch and time domains in real time. As in *pp* where buffers were utilised to apply both transposition and time stretching to a recorded sound, *WarpIn* was used to achieve this in real time. This represents a clear development of the techniques utilised in *pp*. The most clear example of this effect can be found at the end of the piece, where the warp factor is defined by and array of numbers based upon a pelog scale. This leads to 12 warp voices with playback-speed ratios of:

[0.07, 0.1, 0.11, 0.2, 0.33, 0.5, 2, 3, 5, 9, 10, 14]

Equal to a symmetrical transposition in semitones:

[-45.69, -39.86, -38.04, -27.86, -19.02, -12, 12, 19.02, 27.86, 38.04, 39.86, 45.69]

This effect results in the extremely high pitched, almost insect-like sounds appearing at the end of the piece. As the transposition is so radical in both directions, a high pass filter was applied at 20 Hz to reduce any DC offset that may occur from extremely low pitch output. A simpler way to show this effect is again with a spectrogram. Figure 12 shows polyrhythm achieved using the warp effect at the end of the piece. The wood block in this section is playing a constant, mechanical demisemiquaver rhythm with a crescendi/decrescendi. This is an example of how a pitch/tempo transposition can result in an expansion of instrumental voices, often having very different timbral characteristics.

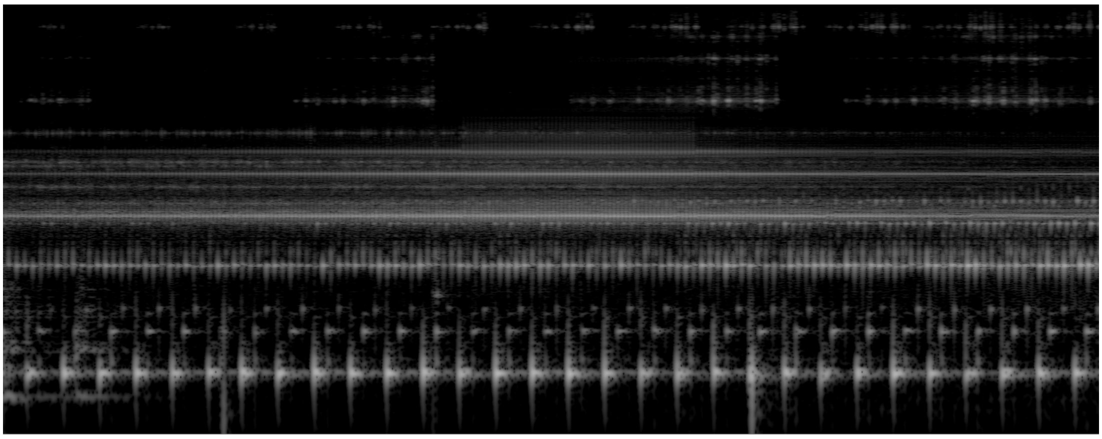


Figure 12: Polyrhythm using warp effect in *For Putten*

In addition to the addition of the warp effect, *For Putten* also improved upon the overall composition system written for *pp* in many ways. One addition that was trialled in this version was that of a user interface. This interface allowed for some simple playback controls and volume to be accessed by the listener. This was then developed and became the user interface which is found in the current framework.

4.3.3 *Stratus* (2012)

Stratus represents the culmination of my approach to (largely constrained) generative music composition. Many of the signal processing elements that appear in the piece are developed versions of prior ideas. Signal processing played a very large role in the aesthetic of the piece, and is relatively complex with relation to the previous two pieces *pp* and *For Putten*. This is largely due to the refined audio routing in the completed framework, which allows for much easier application of serial signal processing and many more processing channels.

As the composition system for *Stratus* has been covered in depth in section 4.1, here the concentration will be on the content of the piece and its individual signal processing elements. Signal processing is slowly introduced to the piece, with the most notably *electronic* sound first occurring through convolution of square-wave and con legno battuto violoncello. This convolution creates an almost 8-bit sounding output and is contrasted with the sound of the trombone and recorded crotales. Here the pulse wave chooses a new note from one of the harmonic series on every new attack from the violoncello. The output is almost reminiscent of the morse code and other data heard over short wave radio frequencies. This is the first time convolution has been approached in my pieces outside of the context of reverberation. As outlined in Emmanuel C. Ifeachor and Barrie W. Jervis (1993), the convolution of two finite and causal sequences $x(n)$ and $h(n)$, of lengths N_1 and N_2 is defined as:

$$y(n) = h(n) \circledast x(n) = \sum_{k=-\infty}^{\infty} h(k)x(n-k) = \sum_{h=0}^{\infty} h(k)x(n-k), n = 0, 1, \dots, (M-1)$$

Convolution as a process takes two input signals and produces a third signal, it is a formal mathematical operation like multiplication or addition. It is used to describe the relationship between three signals of interest: the input signal, the impulse response, and the output signal. In this case the impulse response takes the form of a pulse (square) wave, and the input signal is the con legno battuto violoncello. The output of this convolution is the relationship between these two signals.

The warp effect plays another large role in this piece, providing the same bond between harmony and rhythm that it did in previous work. This is first noticeable early in the piece, where con legno battuto violoncello and crotales are subject to this signal processing. However, the first notable application of this effect is its application to the highly rhythmic trombone and pizzicato violoncello in section six. Here a relatively long attack time was chosen, creating a crescendi which releases throughout section seven. The application of warp in this section creates a dissonant polyrhythm, contrasting with the relatively consonant rhythm of the trombone and cello. However, much of this style of time stretching has already been covered in *For Putten*. One further application of the warp effect that has not yet been approached is to create large soundscapes. Section nine of the piece approaches this effect by chaining the warp synthesiser with a feedback delay. In this section, two violas play the high partials of the union of a harmonic series and another using artificial harmonics. These violas are subject first to the warp to create the chord, then to the feedback delay to create the large "pad" type sound, then to spatialisation to allow for some movement. Aesthetically the goal was to create a lush pad which would move relatively slowly through the multichannel field, creating the illusion of density. The violas are transposed radically over a four octave range to create an extremely large chord, with octave relationships to highlight the original partial. By applying feedback delay to the warp rather than the original sound, it is possible to smear this huge chord across the whole of pitch space. To create a smooth sound five delays are utilised with different delay times but the same amplitude envelope. The sound here performs a crescendi from a relatively thin, dry viola sound into an extremely wide pulsating pad, compounded by the regular bass drum underneath and bass clarinet multiphonics. To further enhance the pad sound it is fed through a reverb processor to smear and soften the sound. The result of this is almost reminiscent of a synthesised sound, giving some foregrounding to the pure sawtooth tone occurring towards the end of the piece.

A further effect which was used quite regularly in *Stratus* was spatialisation. In comparison to the previous pieces, spatialisation in *Stratus* was approached in a much more controllable fashion. The last piece to utilise a spatialisation system was *pp*. In *pp* a sine oscillator was hard coded into the panning system. In addition to this, effects were forced through this in order to be outputted. Not only did this provide some major challenges on the mixing and mastering front, but it also required all panning to be controlled using addition and multiplication factors of a sine oscillator. In *Stratus* this idea was completely scrapped. Now a synthesiser, *Hala*, is used to do this on a completely arbitrary basis. If necessary a sound is routed through this using a bus, rather than forcing all sounds through it. In this approach all sounds are panned conventionally, apart from those which are to be spatialised. Furthermore, the panning algorithm is not hard-coded into the synthesiser itself. This approach uses the monophonic event stream generator `PmonoArtic` as control. This means the synthesiser is created when the `PmonoArtic` is entered, and released when it terminates. Movement is controlled using a standard pattern approach, namely the `Pseg` pattern which allows for a breakpoint envelope style approach to pattern composition. Here the pattern output may represent variations in time that are independent of the rhythmic patterns that express them. The output of `Pseg` is slurred to create an articulation and glissandi between the notes. Spatialisation applied in this way can be heard from the beginning the piece, with granulated air noise from the flute used as the source. Throughout the piece, Brownian motion was used as the primary algorithm for panning in the spatialisation function. This was down to personal preference, and the irregularity of output and ease of control of the algorithm.

The same monophonic event stream technique is used to create glissandi in other aspects of the signal processing. This is first noticeable in section four, with the slowly rising glissandi of the frequency shifter. Applied to the trumpet, this effect is initially quite subtle. Frequency shifting the trumpet leads to a sound almost redolent of the prior saxophone multiphonic. Here the distinction between instrumental and signal processed sounds are blurred, providing an introduction to the highly electronic sounds which occur following this section. Frequency shifting provides an interesting timbre also utilised quite radically in section nine. In this application to the double bass, the frequency shifter is used to highlight upper partials of a fundamental note. The double bass plays the fundamental molto sul ponticello, while the frequency shifter moves between any of 64 harmonics (or subharmonics). This effect is applied using a `Pseg` which allows for glissandi to be applied to both the frequency and rhythm of the frequency shifters movement. It is a good example of the application of the harmonic series to rhythm in the piece. Here both frequency and rhythm are determined by the same harmonic series, only modified to fit within the correct range for the application. In addition to the more notable effects, monophonic event streams are also utilised to provide glissandi in more subtle aspects of the signal processing. For example, by modulating the pitch and time dispersion of pitch shifters the output sound becomes increasingly granulated.

In addition to signal processing, synthesis played a small role in *Stratus*. To generate the output for the final section of the piece, a small additive synthesis operation was applied to a sawtooth oscillator. Here 16 voices are represented by filtered sawtooth oscillators, with low-frequency-oscillator modulated amplitude, filter frequency, and filter bandwidth. Simple modulation of the filter frequency leads to the "waving" through the harmonics. This is intended to show the relationship between the series played on the strings and the fundamental series which the synthesis is playing. A large amount of reverb is applied to the output sound to give it more character. This is the final section of the piece wherein the harmonic series is clearly represented through the first four harmonics played by the string section. The synthesised tone provides some interference with these notes as the strings play varying series' and results in an extremely thick timbre.

5 Non-Realtime Systems

The discussion of the pieces created for the composition portfolio will now be continued. In this section the focus will be on the non-realtime aspects of the systems. Furthermore, he discussion in this section will be approached in a more qualitative context. Non-realtime systems featured heavily in the realisation of the bulk of the compositional work for the portfolio. These systems can have great benefits when composing static pieces. This is highlighted when composing pieces for tape, where much sound processing may go into one small sound-element. In this context, morphing and engineering of multiple sounds can be completed without having to worry about computing constraints. In the context of the acoustic pieces, non-realtime systems are useful in that they allow for discretionary editing. This has been an interesting concept for composers of algorithmic music: whether to edit the final output by hand, or retain the pure algorithmic output. As both techniques have been utilised for this thesis, the results can be evaluated to some degree[51]. In my composition portfolio non-realtime work provided a strong basis for a move toward real time production. As the pieces completed span a number of different techniques, these will be covered in the relevant subsections with appropriate background.

5.1 Sonification

Sonification was the first method used to generate musical data from algorithmic processes. The result was the piece *Warblers*, for two pianos (or ideally, one automatic piano). Techniques learned through the development of *Warblers* paved the way for the later pieces. Indeed, generating data and editing it in a discretionary fashion was the methodology used for all of the instrumental pieces, prior to the development of the SuperCollider system outlined in section 4.1. As the topic of sonification is not covered anywhere else in the thesis, a quick overview of the field will be made before considering the piece.

[51] With regard to my personal composition portfolio, that is.

5.1.1 Background

Research into sonification strives to find an audio representation of complex, multidimensional data (Oded Ben Tal and Jonathan Berger, 2004). Sonification can be used as both a scientific and compositional tool, and thus represents a connection between the two fields. Indeed, much work has been done in specifically translating scientific experiments into music, taking a relatively pure output and representing it through sound synthesis (Sturm, 2001; John Dunn and Mary Anne Clark, 1999; Miranda *et al.*, 2009; Brooks and Ross, 1996). Perhaps the most important aspect of sonification is how the sound is *mapped* to the output. The piece *Warblers* discussed in this section takes a simple one-to-one frequency/amplitude mapping approach. However, more complex systems may be determined for complex multidimensional data in order to represent the character of the data more fully. For instance, one famous example of mapping is present in Charles Dodge's *Earth's Magnetic Field* (1970). For this piece, Dodge uses data from the radiation of the sun on the magnetic field of the Earth. The data from a Bartels diagram showing the fluctuations in the Earth's magnetic field formed the basis for the piece (Doornbusch, 2002). The "sounds of nature" are a focal point for sonificiation, and so some other notable compositions in a similar a vein to Dodge's system can be found in Bob L. Sturm (2005); Anrea Polli (2005).

5.1.2 *Warblers* (2009)

Olivier Messiaen's *Catalogue d'Oiseaux* (1956-1958) represents the main influence for this investigation into sonification as a compositional process. Used in a completely non-realtime fashion, the result was *Warblers*. However, instead of resorting to manual transcription of birdsong, or adapting Messiaen's *style-oiseaux* (David Kraft, 2000; David Morris, 1989), sonfication was utilised. This was in order to capture the different partials involved in the birdsong, the essence of the sound instead of merely the fundamental notes. Chords are therefore apparent which describe clearly the sound, and can be heard in throughout *Warblers*. One example occurs early in the piece, in bars 8-10, and is shown in figure 13.

Figure 13: Birdsong in bars 8-10 of *Warblers*

The system used to generate the base material for the piece was an extremely simple MaxMSP patch. This utilised the `fiddle~` object, and generated a MIDI file with a relatively high resolution from its output. Two files[52] were created using this MaxMSP patch and were initially edited to form a macrostructure in a digital audio workstation. From this point all editing was completed by hand. *Warblers* represents the only foray into sonification for this portfolio, and the output is likely the most highly edited. It can be broken into multiple sections as shown in the score, or into two major sections. These two sections are differentiated by the changing texture of the sound entering the pitch-follower. This is shown later in the piece, where an underlying bias toward the lower register begins. The character of the piece changes substantially at this point. From a relatively placid beginning, an atonal backdrop begins. This is formed from wind-noise entering the pitch-following system. The macrostructure is supposed to highlight this gradual drop in pitch. From the initial playful birdsong in the high registers, through to a noisy section in the lower registers. Birdsong cuts through the noise to form irregular, increasingly agitated phrases. Throughout the piece there is a bias around the tonic of E-flat. This idea for this was inspired by the quick, repeated notes in Maurice Ravel's *Alborada del Gracosio* (1904-05) shown in figure

[52] One from each of the birds sampled: Bonelli's warbler, and Grasshopper warbler.

14[53]. The piece ends on this single note, forming a resolution to the noise that preceded it.

Figure 14: Repeated notes in Maurice Ravel's *Alborada del Gracosio*

5.2 Structures Outside-time

A topic that has not been covered yet, but was utilised prolifically throughout this research is that of outside-time structures. This terminology is taken from Iannis Xenakis, who distinguished three distinct musical structures. These were outside-time, temporal, and inside-time. Outside-time structures represent quantifiable characteristics existing independently of temporal elements; simply, they can be described as *compositional building blocks.* The ordering of outside-time materials into a coherent entity is a function of time, therefore results from the mapping of these elements onto a temporal structure. The temporal structure is based upon rhythmic organisation, the result of this becomes the inside-time structure, representing the completed compositional entity (Flint, 1993). For each of these

[53] Public domain, taken from `http://imslp.org/` accessed: 01/07/12

structures, a kind of algebra was introduced and explained in Xenakis (1992) quoted from (Flint, 1993):

1. The algebra of the components of a sonic event, with its vector language, independent of the procession of time, therefore, an *algebra outside-time*.

2. A *temporal algebra*, which the sonic events create on the axis of metric [measured] time, and which is independent of the vector space.

3. An *algebra in-time*, issuing from the correspondences and functional relations between the elements of the set of vectors X and of the set of metric time, T, independent of the set of X.

For example, a pitch-scale is an outside-time system because no combination of its elements can alter it. The *event*, the actual occurrence of the scale, belongs to the temporal category. A melody or chord on a scale is produced by relating the outside-time category to the temporal category. Both are realisations in-time of outside-time constructions (Xenakis, 1970).

Outside-time structures were utilised throughout all of the compositions with different applications. For example, the prime series in pieces such as *pp* and *Prime Pattern 33*, the Gaussian probability distribution in *Hermit*, or the harmonic series in *Stratus*. Much of the fundamental compositional processes used for the pieces are described in the following section on patterns. However, these outside-time structures provide much of the data used, therefore represent an important part of the essence of the pieces. For example, in the piece *pp*, the prime series from 31 to 71 is defined at the beginning of the system as a global variable. This array is called upon throughout the composition to determine compositional parameters which directly affect the musical output. Fundamentally, the prime series in *pp* stays the same and represents elements on both the macro and micro scale. The array itself is never modified, but it is called upon in different ways utilising methods described in the following section on patterns. This is important because the outside-time structure can be applied on many different levels of composition, from micro to macrostructure - thus affecting the whole piece completely. If one were to modify this variable before runtime, it is likely that it would change the output of the system quite substantially.

5.3 Indeterministic Music

To follow on from the discussion into outside-time structures, how another of Xenakis' concepts influenced the portfolio will be looked at. Specifically, stochastic music and its application in the piece *Hermit* (2011). Stochastic constructs were used widely in the microstructure of many of the compositions in the portfolio. This usually takes the form of a random number generator choosing from a number of possible parameters. However, the piece which embraced this method to the fullest was the short piece *Hermit*. This will now be discussed with regard to the stochastic concepts which were used to generate the material.

5.3.1 Background

Indeterminacy in 20$^{\text{th}}$-century music has taken many forms: John Cage's "chance music" (Miller, 2009), or Karlheinz Stockhausen's "aleatoric music" (Robin Maconie, 1972), or Iannis Xenakis' "stochastic music" (Iannis Xenakis, 1996) are examples of this. Stochastic music was first emerging in the years 1953-55 when Iannis Xenakis began to introduce the theory of probability in musical composition (Serra, 1993; Xenakis, 1966). Stochastic music is based on a system in which the probability of moving from one state to another is defined. The evolution of the process is governed by a weighted randomness, leading to an output which may range from completely deterministic to entirely unpredictable.

In the 1960s, Xenakis began to use computers to automate the stochastic operations needed for his pieces, allowing the computer to make important compositional decisions. During this period Xenakis put forward the idea that stochastic laws could be extended to all levels of composition, including sound production.

> *Any theory or solution given on one level can be assigned to the solution of problems of another level. Thus the solutions in macrocomposition (programmed stochastic mechanisms) can engender simpler and more powerful new perspectives in the shaping of microsounds*

> *than the usual trigonometric functions can [...] All music is thus automatically homogenized and unified.* (Xenakis, 1992, pg. vii)

This concept of stochastic laws controlling compositional micro and macrostructure lead to the creation of his system *GENDY*, discussed previously in section 1.4.2. Much further work has been completed in the field of stochastic music since its inception. Systems such as my own *Stochastic Sampler* and *Stochos* (Bokesoy and Pape, 2003) look at the field of realtime stochastic synthesis, albeit from slightly different angles. Entire books have been written on music and probability (David Temperley, 2007). Stochastic laws are now fully ingrained into algorithmic music through many of the environments used to create it.

5.3.2 *Hermit* (2011)

Possibly the most stylistically stereotypical of the automatic piano pieces, *Hermit* was designed to embrace both the performance medium and the compositional constructs used to the fullest. Written after the longer, arguably more complex *SnSu*, *Hermit* is an exploration into the limits of the methods used to write both pieces. Originally the piece was to be longer, comprising of many sections with different interpolating tempos. This was to be done by rendering multiple MIDI files from Common Music with different parameters, and editing them by hand within an audio workstation. However, after working on this for a while, one of the 60 purely algorithmic takes seemed to have more character than the others and thus became the short piece *Hermit*.

Unashamedly inspired by the works of Conlon Nancarrow, the extremely high speed output and short length are the most obvious features of the piece. Conceptually, the piece is designed to be a decrescendo. A layering of multiple voices at different tempos creates the dense backdrop. Rather than adopting the atonal harmonic context often connoted by stochastic pitch selection methods, notes are selected from a number of pentatonic scales. This leads to a (debatably) consonant, jazz-like output from the system. The 48 scales used were taken directly from Nicolas Slonimsky (1986) “Thesaurus of Scales and Melodic Patterns”. A Gaussian probability distribution was utilised for the pitch/scale selection routine, and many of the other parameters in the system. While the piece does utilise patterns, the defining element of the code is the Gaussian probability distribution. In part, some inspiration was taken from Xenakis' *Herma* (1962) although the methods used were radically different. The overall compositional concept in *Hermit* was to utilise random output, and constrain it using pentatonic scales to produce harmonic output.

5.4 Patterns

The most important method utilised for composition in this thesis, patterns were used in most of the compositions as a fundamental building block. They were used exclusively in multiple compositions as a way to generate material algorithmically. In addition to this, patterns were used as the medium in which to translate the outside-time structure to the temporal structure in the pieces. One of the reasons patterns are so powerful is because it is possible to create extremely complex systems from a relatively small set of basic rules (Heinrich Taube, 1991). One uses patterns in a nested fashion, so that algorithmic constraints may be imposed upon the material from macro to microstructure. This allows for extremely high levels of control, as well as the ability to easily construct complex rulesets from simple building blocks. Patterns were utilised in the composition of systems within both the Common Music and SuperCollider environments.

Patterns have been added within the category of non-realtime systems as fundamentally they exist only in a non-realtime context, they are static. The generative pieces composed using this method of composition are technically performed using event streams. An event is defined in SuperCollider as an environment with a play method[54]. Typically, events consist of a collection of key/value pairs which determine what the play method does. These values can be anything, including functions defined in terms of other named attributes. The modulation of these values using value streams generate the musical output.

This section is used to describe the compositional goals and aesthetic output of the composition portfolio in a more qualitative light. Whereas the concentra-

[54] `http://doc.sccode.org/Classes/Pattern.html/` accessed: 19/06/12

tion in the section on real time systems was on signal processing and technical challenges, here the concentration is upon the overall aesthetic output of the composition itself. This means that it is possible to bundle the static pieces along with the generative pieces and examine each based on its compositional merit (or failings) with regard to the algorithms used to create it.

5.4.1 *Prime Pattern 33* (2009)

One of the earlier piano pieces, *Prime Pattern 33* was the first foray into pattern based composition. In a similar fashion to *Hermit*, the construction concept was to generate multiple files, select those which appeared the closest to my musical intentions, and then edit them to my specification. The code itself is based heavily around the prime number series, and the take eventually selected was number 33. *Prime Pattern* is an example of algorithmic output which was heavily edited in a hands-on fashion within both a MIDI editor, and also a standard score editor. It represents not only the first foray into pattern based composition, but also the first real example of truly algorithmic instrumental output. Whereas *Warblers* concentrated on methods of discretionary constraint, the bulk of the form and character of *Prime Pattern* was determined directly by probabilistic processes in the code. The piece was in essence a stepping stone toward a "pure" algorithmic output.

Due to the reliance on the prime series, many polyrhythms were apparent in the rhythmic output. These polyrhythms added within the code were quantised into semiquavers within a MIDI editor. Further editing within a score editor lead to what was once polyrhythm being transformed into irregular groupings of semiquavers. This helped to retain the original character of the piece while keeping the score relatively simple. Despite the unusual, slightly disjointed phrasing due to the prime series, the piece endeavours to retain an relatively *mezzo* character. Phrase length is determined randomly. This is achieved with a function which returns an array of values which represent the division of a beat. For example, for a septuplet ($\frac{1}{7}$) there may be seven or 14 notes in a phrase; a quintuplet ($\frac{1}{5}$) five or 10. Typically the rhythm array returned for a septuplet would look like:

$$[\frac{1}{7}, \frac{1}{7}, \frac{1}{7}, \frac{1}{7}, \frac{1}{7}, \frac{1}{7}, \frac{1}{7}]$$

This array is read through until the end, creating the desired phrase. The process is executed every time a new rhythm is determined. Dynamics throughout sit within the mezzo-forte and mezzo-piano range, highlighting the intended character of the piece. These were realised largely by hand, however the system had originally chosen them randomly.

Pitch selection in *Prime Pattern* was achieved with two functions. The first dealt with chords, and the second with phrases of single notes. These were triggered using a simple coin-toss method. Using this method the overall probability of a chord being generated was set by hand to just under half that of a phrase. However the basis of the harmony in *Prime Pattern 33* was largely down to a discretionary decision. This took the form of an initial scale, shown in Figure 15. From this scale harmony was derived using the prime series to generate chords and phrases stochastically. For a chord, first the current MIDI note value is passed to the function. To this value, a random prime number in the series up to 13 is added. Then it is decided whether to transpose down an octave or two. This process is repeated either 2, 3, 5, 7, or 10[55] times to create a chord with that number of notes and outputted. A similar process is repeated for the single note function, with some weighting toward ascending phrases.

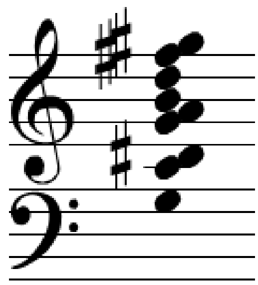


Figure 15: Tone row used in *Prime Pattern 33*

Prime Pattern provided some lessons with regard to the compositional process utilised to generate the piece. These were largely applied in the following piano piece *SnSu*. Primarily, the lesson was that it is extremely difficult to retain complex rhythm when importing from MIDI to a score editor such as Sibelius. This

[55] 10 was used to retain *some* humanisms in the voicing

lead to the decision to quantise the output into semiquavers. However, as the character of the piece was largely retained there was only a minimal aesthetic difference between the quantised and non-quantised versions. After further editing on the score level this was ironed out.

5.4.2 *SnSu* (2011)

Comprised of five smaller sections, *SnSu* or *"Sneaky Suite"* was written in 2011 for automatic piano. In a similar vein to the other pieces developed in Common Music, the concept was to create multiple takes and edit by hand later. Carrying on with the theme of prime numbers, the piece intended to develop further the irregular phrase lengths introduced in *Prime Pattern 33*. This was achieved through some stylistic ideas introduced into the code which are discussed below. The code for *SnSu* is relatively simple, designed to be modified by hand to create different output. The same method to generate the basic material for *Prime Pattern 33* was utilised: a chord function, and a phrase function. These are chosen using a random number generator from a meta-process. In this piece the chord function has an added functionality for the possibility of an arpeggio, increasing the range of the musical output.

In addition to the standard playback functions, a function was added to create scales based upon the prime number series. This takes two arguments, a starting note and an ending note. First, the function chooses a phrase length from an array containing the prime numbers between two and 41. Once this is known, the interval between notes can be formulated based upon the phrase length. The scale becomes the starting note plus the interval until it reaches ending note. In essence this is an interpolation process, the prime series becomes the basis for the rate of change between the starting to the ending note. If the starting note and the ending note are the same, a repeated note will be heard. Floating point values are rounded to the closest MIDI value, or semitone. This system creates the scales used for both the chord and phrase functions, leading to the glissandi-like phrasing heard throughout the piece. In part the idea for this method stemmed from the "arborescences" in Xenakis' music, in this piece these are created using this interpolation/pitch-glissandi function. These "arborescences" are most clearly heard in the first movement of the piece, *Sneaky*, where irregular groups of notes ascending and descending in a glissandi-like fashion.

Rhythm throughout *SnSu* is based largely around the irregular groupings formed from the prime number series. Two sets of rhythms are used, one for the specific rate at which notes will be played, another to decide when to choose another rhythm. Rests are determined using a simple coin-toss system which takes a probability value. If there is no rest then a dynamic from *pp* to *fff* is chosen. This is most clearly seen in the score for the fourth movement of the piece *Undecided Contrast*, where a purer algorithmic output was retained. Other sections of the piece were edited heavily by hand and the indeterministic amplitudes were replaced.

This piece was the last of those written in the Common Music environment. In addition, it represents the last piece to be composed using the technique of cutting and splicing MIDI files together coupled with discretionary editing in a score editor. Compositionally, *SnSu* can be looked at in retrospect as a relatively unsophisticated form. The use of the same fundamental algorithm throughout lead to some repetition in style. Sections *Sneaky*, *Crescendi Glissandi*, and *Double Dip*, for instance, have extremely similar stylistic qualities. In this respect, perhaps there could have been some more musical development on this front. Nevertheless, *SnSu* provided some clear improvements with regard to the algorithms used. Specifically, the use of a meta-process to generate specific musical qualities lead to relatively interesting output before any editing. This concept was developed in later pieces and applied in a more linear realtime context in the SuperCollider environment.

5.4.3 *pp* (2011)

The first piece to utilise patterns in a real time context without any further discretionary editing was the piece *pp*. Patterns were the basis for all algorithmic control in *pp*. The signal processing routines, form, and all piano note production are handled through patterns. This was achieved through a master playback pattern with multiple nested systems within. Each section contains parallel pattern systems for velocity, pitch, duration, and any other parameters pertaining to particular signal processing elements. These pattern systems can be further nested. For instance, one could have a simple sequence and nest a random pattern within which would generate a random number when it was triggered.

Musically, *pp* was an exploration into interweaving acoustic and electronic sounds at a fundamental level. As there is no real *synthesis*, but more a bias toward sound manipulation, the palette of timbre is relevant to the source sound. The initial phrase sets a theme for the piece, this is a good example of the way in which patterns were utilised throughout. Rhythm in this first section of the piece is defined by a nested system of pattern sequences. These sequences are shown on a stave in figure 16.

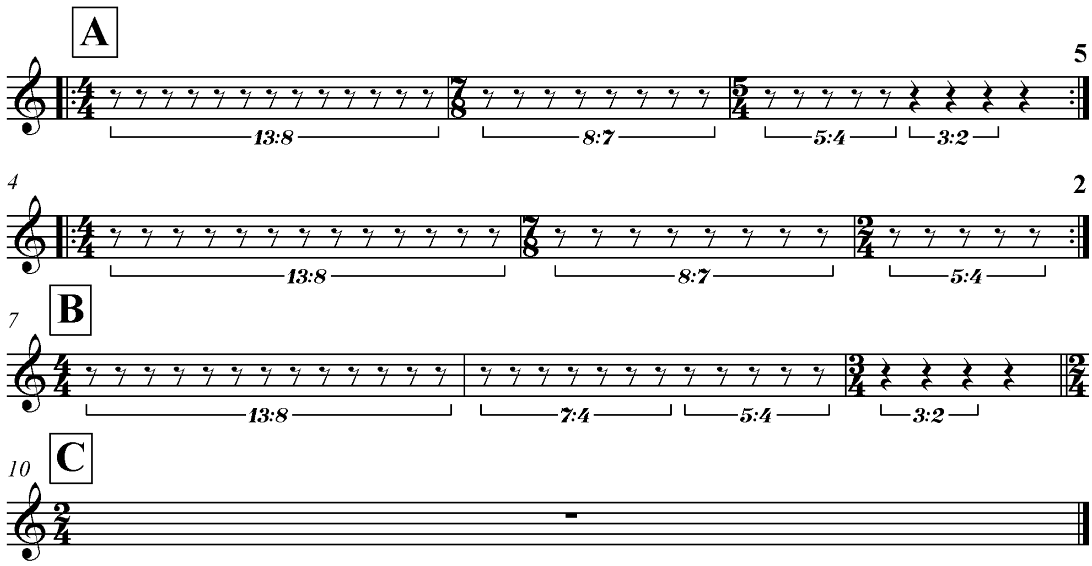


Figure 16: Rhythm patterns in the first section of *pp*

Sequences in this section are chosen using the `Pxrand` pattern system. This means either *A*, *B*, or *C* may be chosen randomly but there will be no repeated sections. Due to this, the section retains some pauses, as when *C* is chosen the system plays a minim. In a similar fashion to the rhythm pattern, note selection is also chosen using a parallel `Pxrand` system. Notes are selected by *folding* the output of a note selection algorithm between two values. This creates the arpeggiated sound. The note selection algorithm takes a note and transposes it by a random prime number from the series. This is then folded between an upper boundary and a lower boundary until a note for playback is found. These boundaries change chromatically to create some movement. Amplitude of the piano is chosen using an exponential distribution between 0.5 and 0.75, leading to a relatively mezzo-forte introduction to the piece. This whole section is recorded into a buffer and re-used later in the piece at different transpositions. As the acoustic and electronic elements of *pp* were designed to work in tandem at a fundamental level, many of the complex timbres throughout the piece are generated from a combination of these elements.

Macrostructure in *pp* was achieved with a simple linear sequencing system. In order to provide some variation, the durations for each section are dynamic. When the system is executed, each section takes a duration from an array of prime numbers. This becomes the section length in seconds. Throughout the piece, certain elements refer to this duration. For instance, the transposition of recorded buffers (and therefore also the rhythm). The concept for this is that the macrostructure, defined by the prime number series, has a direct and immediate relationship with the microstructure. This was to allow the prime series to pervade throughout the piece on every level, a clear example of the use of an outside-time structure in determining the qualities of the resulting temporal musical work.

5.4.4 *For Putten* (2012)

The first piece to utilise the conTimbre sampler, *For Putten* was written in 2012 for percussion. This piece represents a move away from the prime series as the primary outside-time structure, and a move toward musical scale as the primary structure called upon. In this piece the pelog scale is used throughout to determine a number of musical parameters. This was to develop a clear link between pitch and tempo. Additionally, *For Putten* is the first instrumental piece to utilise instruments other than the piano.

Patterns in *For Putten* are slightly more sophisticated than in *pp*, with musical parameters regularly dependent upon each other. An example of this is in section two of the piece, where the duration of a note depends on the instrument played. Here a single pattern 'voice' triggers the sampler. If the bass drum is played then the duration is determined by a weighted random choice between a breve, a semibreve, and a minim. However, if the bongos or congas are played, then the possibilities are a minim, a crotchet, or a quaver. This concept is developed later in the piece in section six. Here the instrument depends on the duration chosen; if a small rhythmic phrase occurs consisting of a quaver-triplet followed by a quaver, the small bongo is played. This rhythmic phrase is approached as a single entity, rather than separate notes. The idea was developed in the piece by making certain parameters of parallel patterns dependent upon each other. For example, in section two the duration parameter is collected from the percussion and fed into three frequency shifters processing the audio signal. This connection means any instrumental percussion event will result in an event from the frequency shifters; new values are chosen from the patterns controlling the frequency shifter parameters on any percussive event. The frequency shifters are therefore dependent upon the percussion to provide rhythmic durations, these could be thought of as triggers in this case.

Musically, one of the key elements of *For Putten* is the use of tremolo. Sampled tremolo is used as a compositional device, contrasting with the highly mechanical rhythms occurring later in the piece. There is a focus upon a perceived dialectic between the the sampled tremolo and the highly mechanical, computer driven performance. This is because each intermingles and interferes with the other, creating a synthesis of both effects. Effects in the piece are designed to form a "glue" between these two sound-worlds: highly mechanical regular rhythms with simple processing, and the irregular sampled tremolo. This is achieved at the beginning of the piece using the pitch shifter to provide processed material as a background to the percussion. A crescendi in the pitch shifter leads to the extreme change in dynamic for the introduction of wood instruments and section three. Instrumentation here follows a form. The introduction utilises only skin instruments, followed by skin and wood, finally a thundersheet is used towards the end of the piece to represent metallic instruments. Frequency and pitch shifters provide some metallic foregrounding in section two, providing a clear ending to the introduction. Overall the introduction to the piece aims to create a tension which is only allowed to release much later in the piece. Release from this tension begins with the consonant quaver rhythm from the wood blocks and kick drum, this is processed and developed into a clearly mechanical but almost organic sounding crescendo at the piece's culmination.

5.4.5 *Stratus* (2012)

Stratus has the most complex orchestration of the pieces. For 21 sampled instruments, it represents the culmination of compositional work completed in generative music. The orchestration itself was inspired by Salvatore Sciarrino's *Luci Mie Traditrici*, and comprises of flute, bass flute, bass clarinet, bassoon, trumpet, tenor trombone, alto saxophone, violin, viola, violoncello, contrabass, crotales, and tam-tam. In part this is a study attempting to draw inspiration from composers with largely contrasting styles.

The piece begins with a pervading aesthetic of fragility, established with extremely high pitched (A5) pianissimo bass clarinet notes interfering with bass flute multiphonics. This use of bass instruments to play high notes was applied throughout the piece to give the note a more strained feeling, as if the player may fail at any point. Clearly since samples are utilised this is never going to happen, this means the note may be held for much longer than would be appropriate for a performer, amplifying the effect. Air noise is also used substantially in the introduction of the piece to further enhance the aesthetic. Granulated and spatialised, subtly foregrounding the impending electronic influence. This introduction is developed in section three with mezzo-piano sul ponticello double bass playing the root note, foregrounding the later sections. Leading from this is a spectrally complex multiphonic from the saxophone and a small crescendi. Here the saxophone multiphonic is paired with the bowed tam-tam, recorded at the beginning of section two and now transposed down.

Rhythm is approached quite subtly during the introduction of the piece, with large sustained notes providing the majority of the output. This is developed through the piece, with complex rhythm first found through the familiar samplerate transposition signal processing applied to the crotales. Later, the tenor trombone introduces the first consonant rhythm, providing an introduction to the highly rhythmic following section. Here the same process of glissandi from section four is applied to the pizzicato strings. Each instrument has a pattern controlling duration and an individual *lag* which is based upon the fundamental harmonic series. To create further rhythmic interest, the harmonic series (as applied to the rhythm) for each instrument is scaled by a different constant. This tapered lag leads to an extremely complex sounding rhythmic output, even though the seed for this is mainly based around minims, crotchets, and breves.

Inspired by the string quartets of Georg-Friedrich Haas, harmony in the piece is determined by a number of harmonic series. However to develop this idea, these are generated on each playback from a common parent. From the initial series of D1, each of the seven following series have a fundamental relating to a note of the original series. The indeterministic, generative element is that on each iteration, the user can never be sure which of the harmonics of D1 will be used to create the other series.

One issue with this is that it is possible to end up with notes that are far outside the range of any instrumental pitch. To remedy this, occasionally the series are transposed down through octaves until a suitable number of notes are within a playback range for that particular instrument. The harmonic series provides the main focus of the piece, and the interaction between these series' becomes the object of study. One of the final sections in the piece shows clearly this interaction between the series, with a synthesised sawtooth wave sweeping through all of the harmonics of the fundamental series, while strings modulate through the others.

An algorithmic approach was also taken towards harmony in certain sections. This was inspired by Xenakis and his application of set theory to sieves (Xenakis, 1992; Xenakis and Rahn, 1990; Ariza, 2005). In this case, the processes of intersection, union, difference, and symmetric difference are applied to multiple harmonic series to create rhythmic and harmonic output. The first example of the application of this in the piece is during the second section of section four. Here the strings join the trumpet and saxophones, opening with a large chord in the same harmonic series. A second chord is found through the symmetric difference of a second series and the first. The symmetric difference of these two series returns a set of of all items which are not elements of both sets. Thus a new scale is created, $seriesA \Delta seriesB$. Mirroring the glissandi of the frequency shifter applied to the trumpet, the strings begin to move toward the second chord according to an interpolation. This creates a highly dissonant number of shifting microtonal chords moving toward an equally dissonant goal.

Aesthetically, much of the piece is rather dissonant and harmony-averse. Perhaps the best example of this is section 11, where excessive pressure is utilised in the string section, and the crotales are bit-crushed and granulated. Here Michael Edwards' technique of Fibonacci transition is utilised to modulate between excessive pressure and standard ordinario playing technique. This is achieved by using the Fibonacci series as probabilities and normalising the result. The part for the first violin is shown in figure 17.

```
// violin 1
    ~delays[11]+0.05,
    Pbind(
        \type, \ctosc,
        \oscout, ~osc_destination,
        \osccmd,
            Pwrand([\noteon,\rest],
                Pseq([Array.fib(8,1,1).reciprocal,
                    Array.fib(8,1,1).reverse.reciprocal
                    ].flop,8).collect(_.normalizeSum),8),
        \voicename, \vi1,
        \midinote,
            Prand(union(~hseries[1],~hseries[3]-48).select({|n,i|
                n>=55}).select({|n,i| n<=84}),inf),
        \dur, Prand((2,4..8),inf),
        \legato, 0.99,
        \amp, Pexprand(0.75,1.0,inf)
    ),
```

Figure 17: Application of Michael Edwards' Fibonacci transition concept

In this code the concept is at work in the key `osccmd`. Here an array within an array is used, the first element contains the reciprocal of the first eight numbers from the Fibonacci series. This is duplicated and reversed to form the second element. The reciprocal is used because it more clearly represents probability. For example, a slowly reducing probability can be represented through the reciprocal of the series:

$$1, \tfrac{1}{2}, \tfrac{1}{3}, \tfrac{1}{5}, \tfrac{1}{8}, \tfrac{1}{13}.$$

Therefore, two arrays of probability remain, one falling from one toward zero, and one rising toward one. These are "flopped" into pairs to match the options (`noteon` or `rest`) and normalised. These pairs are sequenced through and used as the seed for the weighted randomness event stream pattern `Pwrand`. To help visualise this concept, the probability weighting curves are shown in figure 18.

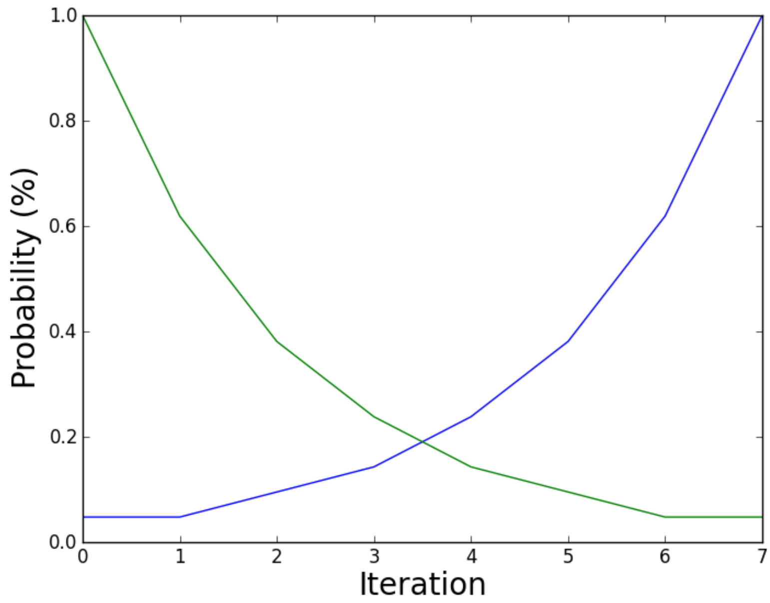


Figure 18: Probability weighting by Fibonacci transition

This is resolved in the final sections where the whole ensemble is utilised to clearly show the series utilised as a whole. Throughout the piece, snippets of each series are shown and modulated with each other often in such a way to cloud the relationship. The end of the piece attempts to clear this and resolve, allowing for all of the utilised series to be represented as large (and often saccharine) chords.

5.5 Tape Music

Tape music was approached only briefly throughout the investigation. As the systems developed concentrate more on reproducing traditional concert music, the tape format was overlooked to some degree. However, two pieces were produced during development of the methodology. These represent some of the work towards a generative system.

5.5.1 *Wet* (2010)

The tape piece *Wet* was written in 2010 for fixed media. The piece was created using ocean noise samples provided by CIBRA[56] and Gianni Pavan. Highlighting the setting of "ocean-noise", at the beginning of the piece a clear dichotomy is established between electronic manipulation and natural source material. Throughout, this dichotomy is broken down, until the source material and electronic manipulation become moulded into one entity. This development of timbre is designed to highlight the underwater setting, wherein methods of synthesis and reverb are used to connote water-like textures.

Many signal processing elements were utilised in the composition of the piece, including some proprietary MaxMSP patches developed for the composition. These patches were used live within Ableton Live by utilising the Max-for-Live system. One may regard *Wet* as a stepping-stone toward a more dynamic system for music production. Previously the concentration was upon fixed media and static pieces, wherein signal processing would be applied to samples and stay the same. Automation was used largely to control signal processing parameters in previous compositions, with some elements controlled by oscillators. However, after *Wet* signal processing would have a dynamic, stochastic influence on the pieces, becoming a *live* element.

Wet is a relatively stereotypical electroacoustic piece, and this was intended from the start. It was an attempt to realise a composition which sounded stereotypically *tape*. One of the main factors of this sound was the large hall reverb, which dominates the piece and adds a sheen to the output. There are admittedly some cliché moments, such as the slow descending glissandi in the beginning of the piece and the over-exaggerated panning. However these add to the character in a certain way and therefore were left. The main musical context was the "un-

[56] `http://www-3.unipv.it/cibra/` accessed: 20/02/2012

derwater" concept, and certain techniques were used to try and promote this. For instance, at the end of the piece (∼3'55) a granular synthesis object was applied to create a very gurgling, squelchy texture to the source samples.

5.5.2 ***Traurig?*** **(2010)**

Another piece heavily influenced by the prime series was *Traurig?*. An earlier piece, and the first composition developed using SuperCollider, it was a combination of discretionary editing and algorithmic work. Structurally, the algorithmic component is based around two major systems. These comprise of: a synthesiser definition, which controls the individual voices, panning, and micro-elements; a task, which triggers the synthesiser definition and concentrates on macro-elements of the piece. The prime series in *Traurig?* is used as a seed for amplitude modulation of synthesiser voices to create layered polyrhythms. This is done using by mixing a random number of voices together, a simple example of this is shown in Figure 19. In this example, the amplitude of two sine oscillators are multiplied by impulse generators at a ratio of 3 : 2.

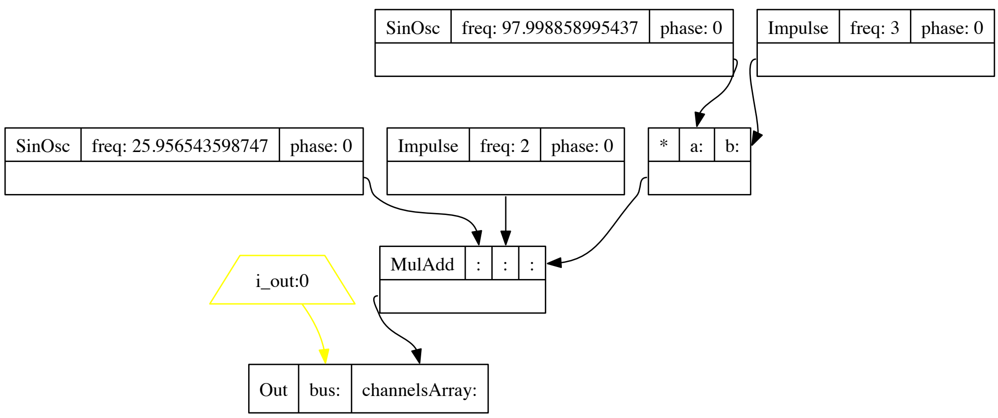


Figure 19: Simple polyrhythm

The number of voices used for each section in the piece ranges from one to seven, this is chosen randomly when the section is triggered by the master task. The rhythm for each voice in *Traurig?* is generated based upon the current section number. This section number then becomes the reciprocal of the equivalent nth prime. A similar process is applied to the voice number, where the voice number also represents the index of a prime in the series. These are multiplied together to create the frequency of the trigger generating the amplitude envelope, thus creating the rhythm for the voice. This is important because of the possible range of these figures. For example, seven voices in section 137 will have percussive frequencies of:

$$[2, 3, 5, 7, 11, 13, 17] * (\tfrac{1}{773}) = [0.003, 0.004, 0.006, 0.009, 0.014, 0.017, 0.022]Hz$$

Whereas seven voices in section 2 will have frequencies of:

$$[2, 3, 5, 7, 11, 13, 17] * (\tfrac{1}{3}) = [0.667, 1, 1.667, 2.333, 3.667, 4.333, 5.667]Hz$$

Traurig? is designed fundamentally to be a piece for *percussion*, the long notes heard are an illusion generated from the release time of the percussive amplitude envelope. Conceptually this was to investigate the perceptible limit of percussive notes. Clearly it is possible to see a deceleration inherent in the system, however a contrary motion occurs in the macrostructure which will now be considered.

Form in *Traurig?* is based upon Conlon Nancarrow's *Canon X* (Kyle Gann, 2006) which occurs in his piece *Study No. 21*. There are two systems, one is accelerating and one is decelerating. This is achieved in *Traurig?* in two ways. Firstly, the deceleration shown above inherent in the note trigger, secondly, a simple reverse operation in an audio workstation. The macrostructure of the system is an acceleration, to create the canon the output audio was duplicated and reversed. This leads to a completely symmetrical form and an interesting timbre change when the two systems approach the same speed. The acceleration of the macrostructure is determined by a simple formula: $t = \frac{p}{i*0.49}$. In this formula t represents the time taken between successive executions of the task; p represents a value chosen randomly from an array of prime numbers from two to 17 (seconds); i represents a counter, increasing with every successive repetition of the task. This generates a simple acceleration; wait time possibilities are shown over 30 task repetitions in figure 20. In this figure, the macrostructure can be seen, with one accelerating and one decelerating set of voices.

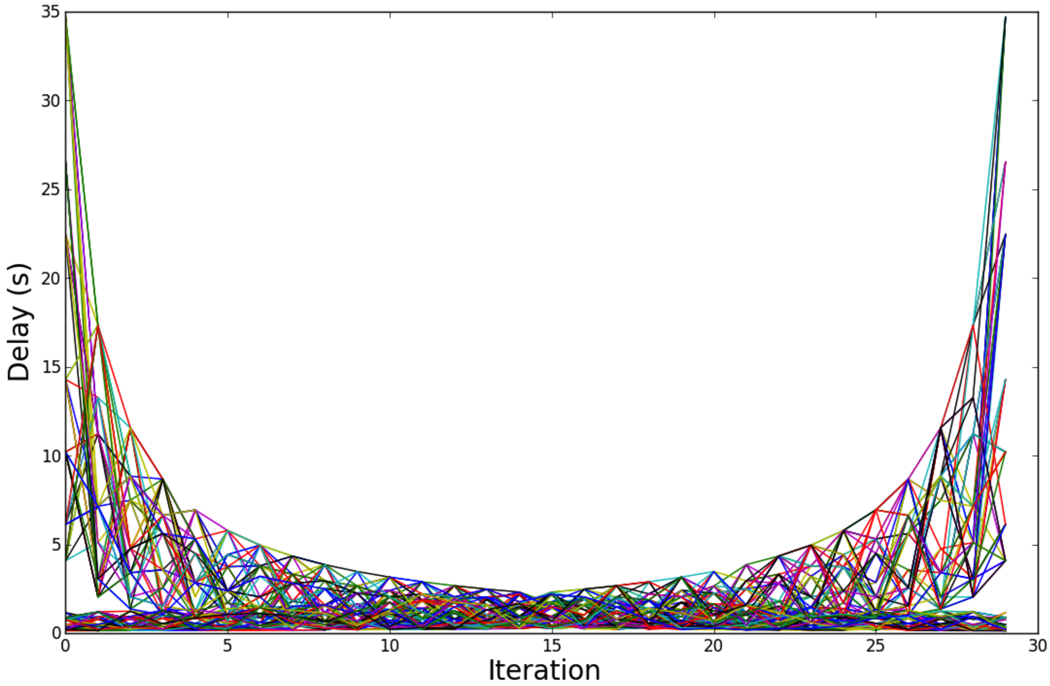


Figure 20: Acceleration in *Traurig?*

Harmony in *Traurig?* is based around chords of five fundamental pitches: G#, C#, F#, G, and C. Harmonic series are then generated with a random number of voices from these fundamentals. The timbre for each voice is chosen from a mix of a number of oscillators: a sawtooth, a variable-duty sawtooth, a variable duty pulse, and a pure pulse with 200 equal-amplitude harmonics. These were used as the source material for the harmonic series due to the rich number of harmonics present in each note, modulating the duty of these waves creates a rich texture. This is enhanced through a virtual-analog feed-forward lowpass filter applied to the output sound.

6 Conclusion

The research completed has covered a wide range of topics. Fields such as digital signal processing, computer science, and music informatics have been touched upon, along with musical composition. Throughout the research my goal has been to find a balance between the technical and the artistic elements of the project. This is with direct reference to the concept of arts and sciences as "alloys" as discussed by Iannis Xenakis in the defense of his own thesis (Iannis Xenakis and Olivier Messiaen, 1985). While the research has gone to the specifics of particular sample interpolation methods in section 3.4, through to a discussion of *style-oiseaux* in section 5.1.2, the thread of musical application has been at the forefront of the discussion. Considering this, and recalling the criteria identified in 1.2.2, reflection upon the work completed can now begin.

6.1 Composition Framework

The composition framework represents the main software achievement from the work completed for this project. The system provides a framework for the composition of generative music, and also an operational delivery methodology. It was developed over the course of the composition of a number of pieces from 2010 to 2012, with compositional workflow and musical output used as a guide for development. Through its development it has been refined into a relatively straightforward system which provides a canvas for generative composition.

As discussed in the previous sections, the main focus was composition through event-driven pattern based techniques. Through the development of this system, generation of large pieces of generative music has become much more streamlined. By facilitating the creation of complex pieces of generative music, it has become the most important piece of the composition process. What makes this piece of software important is the concept of a standardised template for the composition of generative music. What has been created is a template for streamlined workflow, where the composer needs only to write a number of patterns in order to generate a piece of music. Furthermore, the system is not limited to only composition, it is in itself the reproduction medium. Therefore, it is not only a step towards a standardised workflow for generative composition, but also an alternative medium for generative reproduction of music. Perhaps most importantly, delivery of the music created was an important part of the overall package. This was to provide a social context for the system, rather than just a template for composition. It provides a methodology for the delivery of this music[57]. Thus, what begun as a canvas for composition lead to a fully integrated system of composition, "performance", and delivery. This was largely facilitated by the functionality and content already contained within the SuperCollider language.

While the composition framework in its current state is relatively useful, the processes outlined in section 1.2.2 lead to the current implementation. In the case of the framework for composition, the criteria used for the development focused primarily upon the usability factor as explained in 1.2.3. As a composer I required a piece of software that would quickly and efficiently translate my musical ideas into sound, without getting tied up in processes that would disrupt the flow of musical ideas into audio. However, prior to the development of the current system, the music I created was written in a largely non-realtime fashion. Output from the non-realtime systems was then pored over and manipulated by hand to produce the pieces *Warblers*, *SnSu*, and *Prime Pattern 33*. However, this methodology had a number of issues for my way of working. For instance, I was unhappy with much of the original material. In fact, the material that went into the music is likely to be a maximum of about 5% of the generated data. This material had to be generated then listened to in order to determine its musical quality. All of this came before any modification by hand, which is also a long process. Thus the philosophy for advancing my compositional methodology was to spend the time tuning the algorithms, rather than spend it trying to determine the quality of a take and manipulating the output by hand. It could be argued, and it is probably correct that one would become more advanced at writing algorithms for composition, so the time taken afterward would be reduced

[57] This is not strictly part of the composition system, and is discussed in more detail in section 6.3

naturally. However, it is always a four step process of writing, generating, quality control, then discretionary manipulation. Therefore, with *pp* a new methodology was trialed. Here the same pattern-based composition processes that were used in the prior piano pieces were utilised, with more reliance upon a hierarchy of interwoven pattern systems and much tighter constraint upon the probabilistic methods used. By moving to a standalone composition software, where the composer can work with the algorithms as they would with a score, the need for fiddly manipulation of specific notes almost disappears. In essence, the composer can *describe* a section of music using a number of algorithms without having to deal with the underlying data itself. Nevertheless, for all its merits, *pp* and the system used to create it had new idiosyncrasies and bugs that needed ironing out. The main issues with this were the forced-panning algorithm, the focus upon serial signal processing, and the programming style. These are discussed more thoroughly in section 4.3.1. With direct regard to the composition framework, the programming style and layout of the system is what made *For Putten* a leap in progress. By splitting the system up into separate files based upon function, turning the sections into elements of an array, and adding a user interface, the fundamental criteria of usability was vastly improved. With *Stratus* these criteria were again refined, bringing a further drive toward total automation of the system, and further ease of use for the composer.

Nevertheless, there are still many ways in which the software could be developed. On a technical level, it may be advantageous to write the system as a SuperCollider class for increased reusability and further ease of use. This would also allow for the system to be released as a SuperCollider extension, and documentation to be written in the correct format. In addition to this, there are also some avenues of possible development related to the methodology of the composition process. As there has been an overwhelming reliance on pattern framework, an expansion of this framework could be an interesting development. Some of the algorithms used in other elements of the research for this PhD could be translated to a pattern style output. For instance, the cellular automaton used in the *Stochastic Sampler* revision could be applied to a pattern framework to generate event data. Another mode of composition which the system has not approached is that which uses human interaction. This provides a possible avenue for development of the system. Some experiments were made in early revisions for sections to be cued through foot pedal or other human interface device. This could be developed to allow for the system to be operable in a completely live context for true live-electronic concerts. However, this would require some clear structural changes to the system, primarily in the design of the macrostructure. At present all timing relies upon distinct and static section times which are global variables designated on system execution, these would have to be scrapped in favour of a triggering system. Using interface devices such as X/Y pads, accelerometers, or motion detection systems, one could have direct control over specific signal processing applications, or seed them with specific envelopes which would translate through the musical output. This would be quite possible and a way to "humanise" the output if that became a musical context in which to develop the system.

Overall, the framework represents a specific methodology for the composition of generative music. The development of this standardised methodology has resulted in the pieces *pp*, *For Putten*, and *Stratus*. The system will likely undergo further development. However at present it represents a stable, working, environment for the composition of generative music using patterns. By combining this system with the methodology for delivery covered in section 4.2.2, it also represents a way in which to compose, and perform generative music to an audience over the internet.

6.2 Interactive Systems

In the context of the overall research portfolio, the interactive system *Stochastic Sampler* was the major element of the research into signal processing algorithms, and performance-centred systems. The original goal when designing the *Stochastic Sampler* was to create an adaptive, musical, granular synthesis tool. This was approached by coupling a stochastic sampling algorithm with Markov chains. Largely, the inspiration for this project was from Iannis Xenakis' work with Markov chains, and Henry Cowell's concept of interval-tempo relationships. The *Stochastic Sampler* was to be a combination of these two approaches. The output of the system was not only intended to achieve a level of interactivity along similar lines as Robert Rowe's *Cypher* or George Lewis' *Voyager*, but also a flavour of output more indicative of a lower level synthesis approach. This concept of creating a system which not only attains musicality through interactive phrase-like sampling and playback, but also through a more high speed 'smeared' output, is what gives the *Stochastic Sampler* its unique output. What makes this important and different from other systems is that instead of using a static algorithm to determine output pitch or tempo, Markov chains allow the system to reference past musical material interactively.

The development of the *Stochastic Sampler* had to consider all of the criteria introduced in section 1.2.3. This is because in the development of the system, there were a number of technical and aesthetic considerations which needed to be looked at. For instance, In the initial MaxMSP system, the method in which the Markov chain was used in the control method meant that for each voice a new probability table was being created. Furthermore, the system was inefficient, the interface was not particularly friendly, and it was slow to react musically on initialisation. There were issues with the usability, musicality, and efficiency of the system. However, these issues spurred the development of the Java/MXJ system. Through reflection upon the problems of the prior system, the Java version was to fix many of these problems. However, while this revision achieved significantly better efficiency, the actual implementation had a number of issues which rendered it unsuitable for musical application. This was largely down to bugs in the Java code. Furthermore, the Java version now did not have any place within a system for composition designed in SuperCollider. Therefore, through careful consideration of what I wanted the system to actually achieve, the final revision of the system was created in SuperCollider. This version alleviated the problems with musical output in the Java system, provided a very simple method in which to interact with the system, integrated with the composition environment, and was extremely efficient. The process for the development of the final revision started with a reflection upon the problems with the prior versions. The Java system was too complex, buggy, and did not fit with my current composition framework; the MaxMSP system was too inefficient for use in a composition. After considering this, it was determined that the SuperCollider revision had to fit into and work with a system for composition just as any other signal processing tool would. Therefore it was developed as a SuperCollider class, with specific parameters that could be modified or automated using a standard pattern approach. In addition to this, the efficiency of the system had to be increased radically for it to operate in this way. By carefully considering how the signal processing system would operate, this was achieved. Therefore, while the initial MaxMSP implementation, and the following Java revision had significant faults when considering their realtime compositional application, through the methodology of reflection and revision as outlined in section 1.2.2, the SuperCollider system was developed. For further details on the specifics of this system, see section 3.5.1.

However, the *Stochastic Sampler* was not the only outcome of my research into granular synthesis techniques. One aspect of the system which facilitated further research was the separation of control and player modules completely. This meant many different approaches could be attempted merely by interfacing with the existing system. The primary alternative control mechanism came in the form of cellular automata. Cellular automata were approached as an investigation into the sonification of complex algorithmic data. These kinds of deterministic systems were not approached again in any depth throughout the portfolio. A very similar approach had already been made by Eduardo Miranda (Miranda, 1995). However, the approach made for this research differed in some significant ways. Specifically and perhaps most importantly, mapping of the data, and source audio. The specifics of our respective mapping approaches were covered in section 3.5.2. Mapping is important because in a system such as this, the mapping strategy can largely determine the output sound. In addition to this, in my approach the control and player mechanisms are separated; the output is not only determined by the automaton but also by the sampling mechanism. By using an individual mapping strategy and incorporating an further layers of complexity in the form of probability-derived sampling and live input audio, the complexity of the audio output is increased.

As the *CA Sampler* followed a similar revision strategy to the *Stochastic Sampler*, similar conclusions can be drawn. From a personal viewpoint, in both cases the original system was more successful than the Java version which supersedes it. However, the SuperCollider systems that form the final revision of my investigation to granular synthesis during this project are a large leap in all criteria. For example, with regard the *CA Sampler*, the original system operates with a lack of efficiency in MaxMSP, however, outputs what I personally consider to be interesting audio with relevance to the automaton. The SuperCollider system performs the same task with the ability to be used as a signal processing tool, or as a standalone system. It is thus forms one "building block" of signal processing tools that may be called upon during composition. It not only facilitates enhanced usability through integration with the composition system, but also musicality through this, and also efficiency due to a more advanced implementation.

Another investigation which stemmed from the work with the *Stochastic Sampler* system, was that of spatialisation through dynamic systems. By extracting parameter data from the from the control system and applying statistical formulae to it, the audio output was linked to the parameters of the dynamic spatialisation system. This provided a unique way of linking the musical output completely coherently with the spatialisation. Craig Reynolds' *Boids* system was chosen because the concept of many separate *boids* fits with the multiple Markov chain voices from the audio system very clearly. Here a parallel can be drawn: each voice has a clear point in space, but a collective algorithm controls the whole swarm, just as the performer controls the whole Markov system. In essence the performer conducts the swarm, his musical actions lead to interactive reactions from both the audio and the spatialisation system. This is important because it clearly and coherently merges the two systems into one musical entity. The system of spatialisation embarked upon when the *Boids* system was applied to ambisonics was one that was not looked at again until the end of the project. After considering the usability criteria, the system was then revised in SuperCollider as a class. Much like the *Stochastic Sampler*, this meant it could be used quite simply in any composition process embarked upon using my framework.

There are still some areas in which the system may be improved. The sampling system is hinged upon an application of probability to core function. This is an area which could be developed, as it is a rather basic approach to the problem. In essence, a probability is used to determine whether the system will record or play back on any given trigger. Developing this system may lead to a great change in the musicality of the output. Furthermore, there are some issues[58] with the current Java implementation which would be ironed out with a rewrite. These were solved through the rewrite of the system in SuperCollider[59], however it may be advantageous to have a MaxMSP/Java system on the same par as the SuperCollider system.

6.3 Performance and Delivery

The technical aspects of performance and delivery have been discussed previously with regard to the composition framework. However some concepts were touched upon throughout section 2 which can now be considered. One of the main arguments stated about the consumption of music in contemporary society, was that the 'aura' is compromised in a recorded musical work. Therefore it could be asserted that given that the majority of the work completed for this portfolio is based on sampled instruments, there is a contradiction in terms. The key point here is that the portfolio is a compilation of *instrumental pieces* performed mechanically[60]. Aesthetically, the decision was made to utilise standard instrumental sounds rather than synthesis. This decision was made to provide a clear parallel with traditional concert music: pieces for tape were created, however the emphasis has been on classical orchestration. The sampler was used to represent the instruments due to my perception that there is a current lack of a feasible alternative. Perhaps at this point it is worth reflecting on the other approaches

[58] Currently the autosampler Java class has an intermittent buzzing sound in the output, and issues with sound quality.

[59] See section 3.5.1

[60] With *Traurig?* and *Wet* as the exceptions.

available, in order to appraise how the decision to use a sampler-based system was settled upon. There are two other possibilities that could potentially improve upon the current system of sampled instruments. These are robotics and physical modelling. However, these methods have their own major weaknesses which sampling bypasses. With regard to robotics the problems are financial and technical. As this research was focused upon composition and software development, rather than mechanical engineering, the decision was made to leave this idea and use the time for research into more relevant fields. In addition to this, the cost of assembling a number of instruments would become a largely immovable obstacle. Physical modelling has some fantastic possibilities, and clear advantages over sampling in some regards. For example, one advantage of a physical modelling approach is that when a note is played at a specific dynamic repeatedly, each instance will have its own individual characteristics. The same cannot be said for sampling, where some homogenisation of timbre is natural due to the static source material. However, the palette of timbre available *overall* in physical modelling pales in comparison to the ∼86,000 samples available in a library such as conTimbre. For complex pieces such as *Stratus*, a sampling approach is (in my opinion) the next best thing to a real orchestra at this point in time. With reference to the criteria defined in section 1.2.3, the decision to retain a sample-based approach rather than the alternatives discussed could be attributed to usability. In this case, it is more a question of feasibility. During the period of research I had extremely limited access to physically modeled or robotic instruments. Due to the realtime composition methodology used, composing for these instruments was not a feasible solution. Therefore, the most *usable* approach was that of a sample-based system.

Regarding delivery, as discussed in section 4.2.2, the importance of this topic came through reflection upon the social context of the music being produced. The major question that had to be asked was, what use is music without anybody to listen to it? Therefore a delivery methodology had to be created which allowed the algorithms and music produced through the research period to find a social context. However, there are already a number of systems in operation which deliver music using a digital medium over the internet successfully. Examples of these have been discussed in section 4.2.2. Through the consideration and testing of various approaches to the delivery of generative music in the aforementioned section, the most viable approach found was the already widely-used method of streaming internet radio. By utilising a server system such as Icecast, a stream can be set up simply by anyone with a personal computer. What is most appealing about this approach is that the listener does not require any complex software or have to deal with anything out of the norm. Indeed, the stream can be accessed by a web browser and many standard players have the functionality built in as standard. The other approach tested was a system in which a user could utilise software written in MaxMSP to request a piece to be generated on the fly. This however leads to major problems with concurrency, as no more than one user can request a piece to be generated at any one time. Delivery strategies for the musical material was approached as an important contextual obstacle that needed to be overcome throughout this research. As the audience in this case would usually be in their homes, a strategy which allows for as many devices as possible to access the stream is important. What is clearly advantageous about the internet radio approach is that it expands the potential userbase from those at their computers, to anyone with a client that can decode an online MP3 stream. For example, this could be a smartphone or similar device with internet connectivity. This is important because it means the audio stream can be decoded not only in the home, but anywhere a 3/4G connection can be obtained. Therefore, this delivery approach exceeds the original goal of developing a system for the delivery of music to the home listener. By removing the requirement for physical media or software installation, and allowing for the user to connect using familiar software, a much wider userbase is attainable through already established means. By looking back at the criteria discussed in the introduction it's clear that from the perspective of the user, here the major factors are usability and efficiency. The other systems discussed in section 4.2.2 are let down at present by their current implementation, and the reliance upon third party software. By using an internet radio approach which is accessible by most modern digital devices, without any third party software, the music is available to more people. It is an efficient method of music delivery, which retains the qualities of generative music while finding a middle ground for usability from the perspective of the user.

6.4 Compositional Outcomes

The portfolio of composition is in essence a study in the refinement of the use of patterns for composition. From the broad algorithmic strokes representing the sections of *SnSu*, through to the extremely specific systems of *Stratus*, patterns have been at the heart of many of the pieces. On a broader level, throughout the research period the musical output has been a direct representation of the software systems created. There was a clear direction in mind, towards a generative system which could generate music with no human intervention. This is an element of the compositional work that had a clear migration during the research period, broadly between static pieces and generative pieces. Originally, notation would be edited discretionarily after the raw data was generated by an algorithm. Towards the end of the musical output this system was handled completely by the generative system, with no discretionary editing necessary. This was possible by working very precisely with parameter constraints pertaining to the musical output. By retaining strict compositional control over the variance of the output, this generative methodology attempted to achieve spontaneity while not sacrificing aesthetic quality. Here the methodology of composition is important. As the composer has total control over the variance of the output for any specific element of the piece, generative elements can be finely tuned. This means that it is very easy for a composer to create a piece with clear individual character, while retaining desirable generative qualities.

The earlier pieces fall mainly into the category of "static" algorithmic pieces which were composed from a top-down perspective. That is, the algorithms were designed quite loosely with quite a lot of variation in mind. This is most apparent in the early pieces for two pianos. Here the details of the piece, the intricacies and nuance were composed by hand. The method here was to generate a (sometimes large) number of individual takes and qualitatively assess this, deciding which would be used for the final production. After this process large chunks would be joined together in a MIDI sequencing system, and edited on a macro-scale. From there a score editor was used by hand to work with the piece until it was deemed acceptable for release. One exception to this trend was the piece *Traurig*, which was the first example of a piece which retained much of the original algorithmic output with only some macro-structural editing in an audio workstation. This earlier methodology was gradually refined through the composition of a number of pieces. Many pieces followed the pieces for two pianos which did not make it into the final portfolio. These were largely electroacoustic works designed in an audio workstation, with MaxMSP processing "printed" onto certain sections of sound. Workflow in this system was difficult to manage. In addition to this, the large amount of various samples in the pool in different stages of affected sound lead to an over-complicated system. Furthermore, with the large amount of processing required it seemed that using the audio workstation in this way was providing more problems than it was solving. Therefore, a system was designed in MaxMSP to attempt to alleviate some of these problems, with much the same concept in mind as the final SuperCollider framework. However, this was later abandoned due to my personal preference for the way processing and composition is managed in the SuperCollider environment. The exception to this was the piece *Wet*, which was designed as an electroacoustic piece within an audio workstation, however used Max-for-Live realtime processing throughout.

With the development of the composition methodology in SuperCollider, a median was found between algorithmic constraint and compositional control. In this context these could even be considered as mutually equivalent, as nearly all of the parameters are controlled with one algorithm or another. However there are some considerations to be made as to the result of the methodology utilised. The three main generative pieces, *pp*, *For Putten*, and *Stratus* are all completely linear in design. That is, the macrostructure of the piece never really changes. In fact, the constraint to which the algorithms operate is so narrow that a listener may not even notice that specific details are stochastic by nature unless listening carefully. Even though some rather large concepts are tacked with an algorithmic approach, such as harmony in *Stratus*, or vast transpositions and section length in *pp*, the character remains the same. Primarily this decision was made to allow for a concise aesthetic, without too many algorithmic surprises that may lead to poor quality representation. The algorithmic variation between performances with this approach are akin to a symphony orchestra playing the same piece twice. This is key to the research project as it represents a clear and practical solution to the issues raised regarding fixed media in section 2.2.1. Due to these variations in performance attributes on each reproduction of the pieces, the context for consumption must be one which is able to retain these qualities.

6.5 Research Outcomes

Here the outcomes of this thesis can be addressed in a direct manner, with regard to the goals stated in section 1.2.1. Throughout the thesis there have been many different threads of thought which have referred back to these goals, and have been developed with accordance to the criteria and methodology noted in the introduction. These threads will now be brought together in order to clearly address the achievements, or the failings of the approach made towards furthering this field.

To create a system for generative "performance" of concert music for the home listener. In order to start generating musical works which fell between the constraints set by the research proposal, the system with which the music was to be created had to be written. The goal was to create a system for composition and reproduction of music which retained specific attributes of performance. Furthermore, my own experience of composition lead to the idea that a system that produced music in this way would have to have a social context. In essence, it would have to be able to deliver this music to an audience while retaining the performance attributes that were sought out during the realisation of the system. Research found that over differing continents and varying spans of time that the classical music concert is dropping in popularity, and personal listeners are taking the majority of the market[61] (Tak Wing Chan and John

[61] An argument could be made that a new survey of personal listener responses is a require-

H. Goldthorpe, 2007; Henk Roose and Alexander Vander Stichele, 2010; Noriko Manabe, 2008; Bonita M. Kolb, 2001). Therefore it was chosen that the system would be developed with this social context in mind. After much research into various methods of realtime and non-realtime composition through multiple environments and modes of realisation, the pattern style event-based system was chosen as it fitted into my composition methodology well. The reason for this choice was largely due to the flexibility of patterns. As they can be nested within each other, extremely complex systems can be created from basic building blocks. This allows for extremely precise parameters, or large swathes of material to be controlled all using the pattern methodology. In order to find the most applicable source of audio data much time was spent researching possible opportunities. The core opportunities for this realisation were based largely on samplers, physical modelling techniques, and robotic opportunities. Through this research I became involved directly with the development of conTimbre, a large sampler and sample library developed exclusively for composers of contemporary music. A point should be made here that I believe a sampler can never be a perfect solution, however at this point in technology it represents (for the goals of this research) the best solution short of human performers.

To develop this system to incorporate a feasible delivery method. Chapter 4.2.2 details the core working of the delivery method with multiple systems trialed and considered with their respective pros and cons in mind. Along with this it discusses how the concept of "delivery" became a large part of the research project, and why it is important. Various other systems have been implemented already to achieve a similar goal, this approach was intended to be a development of these. Through this research I intend to pave the way for further research, by examining various techniques of music delivery and looking at the problems and advantages of each. A system was developed that would provide perfect functionality for a single user, however it fell flat when concurrency was considered. A view toward previous work and already operating solutions lead to

ment for this assertion. However, this research project is clearly placed in the field of creative music practice. Therefore a survey of this sort would be out of scope, but could be considered for further work.

a much more concise and easy to achieve solution: radio. This approach focused on as a composer would be able to use existing tools and tested methodology to broadcast a "concert" program relatively simply. Using this system allows for any potential listeners to tune in with software they likely already have, such as Windows Media Player, or iTunes. As this concept removes the difficult problem of concurrency, as well as the issues with mixing and mastering, while allowing for a generative broadcast to be achieved, it is the preferred method of delivery.

To create a portfolio of work for such a system. The work completed spans a large range of instrumentation and various compositional approaches. Compositionally, I would hope that the output of the musical work represents not only a maturity in technical matters but also one in composition itself. Through development of conTimbre the instrumentation opportunities available were raised exponentially. This provides a fantastic opportunity to develop further works within the framework and continue to refine the musical output in future.

To contextualise this concept within contemporary society. Primarily the social context approached for this system was the internet. In some ways this could be regarded as an easy way out, as at present the social media bubble seems to be at its peak... However, there are continuing examples of how music is shared within these communities, and how musicians and composers as a group interact and use these services. Services such as SoundCloud allow for composers to upload their musical works and have a platform for public listening. This combined with large and dedicated groups of researchers and programmers of musical languages such as SuperCollider or MaxMSP lead to an interesting platform for musical delivery. Arguably a paradigm shift between the concert setting and record industry dominated delivery stereotype of yesteryear, internet based social media is clearly a viable way to deliver music to a large audience. This is proven through numerous and continuing success stories of internet musicians.[62] Nevertheless, the key concept is that like minded groups of composers

[62] The validity of these stories is likely debatable, however their occurrence seems to be increasing in regularity

and developers can work together and provide a social context for the music they create. Therefore, proposed in this thesis was a social media approach which would combine commercial concepts like SoundCloud (sharing audio files), with those like Sccode.org[63] (sharing code), and Bemmu & Viznut's online system for creating music from short C programs[64] (online interpreter operating through web browser). This provides a clear area for further work with the possibility of an interesting and significant development in the contemporary generative music community. My contribution to this concept is an alternative means and methodology for the creation and delivery of generative media to a target audience. By combining the composition approach used with existing radio broadcast software, generative pieces can be developed and delivered to an audience practically.

To challenge current idiosyncrasies of recorded media and provide an alternative. Through research into the effect of recorded media and its perception in section 2, selected ways in which the output media could be developed were investigated. Many of the concepts investigated with regard to recorded media's faults can be regarded as relatively contentious. However it is necessary to approach this with a view to improving what must be a system with faults, otherwise there would be no need to improve it. Much work was drawn from the concept of improving the listeners experience of music they listen to at home. The main issue approached was that there are no surprises in recorded media, the piece was only performed once: nothing changes. Here we approach this problem by using algorithms to subtly change the way a piece is produced on each playback. This provides a clear alternative to concert performance and recording. There are some sacrifices, specifically some aesthetic and timbral quality through the nature of the source material, but the advantages gained are the qualities inherent with dynamic performance. Indeed, this production methodology is not intended to be a replacement at this point in time, but an alternative.

To work within the field of generative music and create a formalised, classically contextualised output. Generative music as a field appears to be

[63] `http://www.sccode.org/` accessed: 01/09/12
[64] `http://www.bemmu.com/music/index.html/` accessed:01/09/12

relatively niche, even with mainstream artists such as Brian Eno and Autechre utilising this approach. This may be a result of an over emphasis on pieces which are not really designed to be concert music but rather art installation type systems. My goal was to create a body of work developed algorithmically in real time, but was not instantly identifiable as “generative”. The body of work was to be contextualised within the genre of contemporary classical music, not primarily algorithmic or generative music. Primarily the goal was always to create meaningful output, to contextualise this within society, and further the field of generative music by some degree. What I have done is create a body of musical work which was created through algorithmic and generative means, while in parallel developing a system and methodology for the production and delivery of this music.

References

Ajay Kapur. A History of Robotic Musical Instruments. In *Proceedings of the International Computer Music Conference*, 2005.

Alexis Kirke and Eduardo Reck Miranda. A Survey of Computer Systems for Expressive Music Performance. *ACM Computing Surveys*, 42(1), December 2009.

Alf Gabrielsson and Siv Lindström Wik. Strong experiences related to music: A descriptive system. *Musicae Scientiae*, 7(2):157–217, 2003.

Alice V. Clark. Listening to Machaut's Motets. *The Journal of Musicology*, 21 (4):487–513, 2004.

Charles Ames. Automated Composition in Retrospect: 1956-1986. *Leonardo*, 20 (2):169–185, 1987.

Charles Ames. The Markov Process as a Compositional Model: A Survey and Tutorial. *Leonardo*, 22(2):175–187, 1989.

Andrew Hugill. Internet Music: An introduction. *Contemporary Music Review*, 24(6):429–437, 2005a.

Andrew Hugill. From online symphonies to MusiMOO: Some creative uses of the internet in music and education. *Contemporary Music Review*, 24(6):527–540, 2005b.

Anna Sophie Christiansen. Beyond the Charts: MP3 and the Digital Music Revolution. *Computer Music Journal*, 25(1):64–66, 2001.

Anrea Polli. "Atmospherics/Weather Works": A Spatialized Meteorological Data Sonification Project. *Leonardo*, 38(1):31–36, 2005.

Christopher Ariza. The Xenakis Sieve as Object: A New Model and a Complete Implementation. *Computer Music Journal*, 29(2):40–60, 2005.

Christopher Ariza. Automata Bending: Applications of Dynamic Mutation and Dynamic Rules in Modular One-Dimensional Cellular Automata. *Computer Music Journal*, 31(1):29–49, 2007.

Philip Auslander. Lucille Meets GuitarBot: Instrumentality, Agency, and Technology in Musical Performance. *Theatre Journal*, 61(4):603–616, 2009.

Bhesham R. Sharma. *The Death of Art.* University Press of America, 2006.

Bob Katz. *Mastering Audio: The Art and the Science.* Focal Press, 2003.

Bob L. Sturm. Pulse of an Ocean: Sonification of Ocean Buoy Data. *Leonardo*, 38(2):143–149, 2005.

Sinan Bokesoy and Gerard Pape. Stochos: Software for Real-Time Synthesis of Stochastic Music. *Computer Music Journal*, 27(3):33–43, 2003.

Bonita M. Kolb. The effect of generational change on classical music concert attendance and orchestras' responses in the UK and US. *Cultural Trends*, 11 (41):1–35, 2001.

Brian Eno. Generative Music, June 1996. URL `http://www.inmotionmagazine.com/eno1.html`.

Stephen Brooks and Brian J. Ross. Automated Composition from Computer Models of Biological Behaviour. *Leonardo Music Journal*, 6:27–31, 1996.

Ludger Brummer, Guenther Rabl, Konrad Boehmer, Jean-Claude Risset, Jonty Harrison, Francois Bayle, Johannes Goebel, Francis Dhomont, and Karlheinz Stockhausen. Is Tape Music Obsolete? Is Spatialisation Superficial? *Computer Music Journal*, 25(4):5–11, 2001.

Carroll C. Pratt. The Spatial Character of High and Low Tones. *Journal of Experimental Psychology*, 13(4):278–285, 1930.

Jaques Chareyon. Digital Synthesis of Self-Modifying Waveforms by Means of Linear Automata. *Computer Music Journal*, 14(4):25–41, 1990.

Charles Amirkhanian. Interview with Composer Conlon Nancarrow. *Selected Studies for Player Piano*, (15).

Charles W. Warren. Brunelleschi's Dome and Dufay's Motet. *The Musical Quarterly*, 59(1):92–105, 1973.

Christopher Ariza. Two Pioneering Projects from the Early History of Computer-Aided-Algorithmic Composition. *Computer Music Journal*, 35(3):40–56, 2011.

Nick Collins. The Analysis of Generative Music Programs. *Organised Sound*, 13 (3):237–248, 2008.

Craig Wright. Dufay's "Nuper rosarum flores", King Solomon's Temple, and the Veneration of the Virgin. *Journal of the American Musicological Society*, 47 (3):395–427+429–441, 1994.

Curtis Roads. *The Computer Music Tutorial*. The MIT Press, 1996.

Dave Burraston and Ernest Edmonds. Cellular automata in generative electronic music and sonic art: a historical and technical review. *Digital Creativity*, 16 (3):165–185, 2005.

David Kraft. *Birdsong in the music of Olivier Messiaen*. PhD thesis, Middlesex University, 2000.

David Morris. A Semiotic Investigation of Messiaen's 'Abime des Oiseaux'. *Music Analysis*, 8(1-2):125–158, 1989.

David Psenicka. Automatic Score Generation with FOMUS. In *Proceedings of the International Computer Music Conference*, 2009.

David Temperley. *Music and Probability*. The MIT Press, 2007.

Denis Harbinson. Isorhythmic Technique in the Early Motet. *Music and Letters*, 47(2):100–109, 1966.

Paul Doornbusch. Composers' views on mapping in algorithmic composition. *Organised Sound*, 7(2):145–156, 2002.

Douglas Bartholomew. Preamble to a phenomenology of music. *Journal of Musicological Research*, 5(4):319–360, 1985.

Eric Drott. Conlon Nancarrow and the Technological Sublime. *American Music*, 22(4):533–563, 2004.

Edgard Varèse. Varèse envisions 'Space' Symphonies. *New York Times*, page 7, December 1936.

Edmund Husserl. *The Phenomenology of Internal Time-Consciousness*. Indiana University Press, 1964.

Edward A. Lippman. Spatial Perception and Physical Location as Factors in Music. *Acta Musicologica*, 35(1):24–34, 1963.

Emmanuel C. Ifeachor and Barrie W. Jervis. *Digital Signal Processing: A Practical Approach*. Pearson Education Limited, 1993.

Ellen Rennie Flint. Metabolae, Arborescences and the Reconstruction of Time in Iannis Xenakis' Psappha. *Contemporary Music Review*, 7(2):221–248, 1993.

Denis Gabor. Acoustical Quanta and the Theory of Hearing. *Nature*, 159:591–594, 1947.

Gary S. Kendall. Spatial Perception and Cognition in Multichannel Audio for Electroacoustic Music. *Organised Sound*, 15(3):228–238, 2010.

Gary Zabel. Adorno on Music: A Reconsideration. *The Musical Times*, 130 (1754):198–201, 1989.

Gottfried Michael Koenig. Project 1. *Electronic Music Reports*, 1(2):32, 1970.

Gunter Kreutz, Emery Schubert, and Laura A. Mitchell. Cognitive Styles of Music Listening. *Music Perception: An Interdisciplinary Journal*, 26(1):57–73, 2008.

Maria Anna Harley. An American in Space: Henry Brant's "Spatial Music". *American Music*, 15(1):70–92, 1997.

Harvey Fletcher and W. A. Munson. Loudness, Its Definition, Measurement and Calculation. *Journal of the Acoustical Society of America*, 5(2):82–172, 1933.

Frederik Hedelin. Formalising Form: An Alternative Approach to Algorithmic Composition. *Organised Sound*, 13(3):249–257, 2008.

Heinrich K. Taube. *Notes from the Metalevel*. Taylor and Francis Group, 2004.

Heinrich Taube. Common Music: A Music Composition Language in Common Lisp and CLOS. *Computer Music Journal*, 15(2), 1991.

Henk Roose and Alexander Vander Stichele. Living Room vs. Concert Hall: Patterns of Music Consumption in Flanders. *Social Forces*, 89(1):185–208, 2010.

Henry Brant. The Uses of Antiphonal Distribution and Polyphony of Tempi in Composing. *Bulletin of American Composers Alliance*, 4(3):13–15, 1955.

Henry Cowell. *New Musical Resources*. Cambridge University Press, 3 edition, 1996.

Lejaren A. Hiller and Robert A. Baker. Computer Cantata: A Study in Compositional Method. *Perspectives of New Music*, 3(1):62–90, 1964.

Peter Hoffmann. Towards an "Automated Art": Algorithmic Processes in Xenakis' Compositions. *Contemporary Music Review*, 21(2/3):121–131, 2002.

James Hooper, editor. *Peirce on Signs: Writings on Semiotic by Charles Saunders Peirce*. The University of North Carolina Press, 1991.

Iannis Xenakis. Determinacy and Indeterminacy. *Organised Sound*, 1(3):143–155, 1996.

Iannis Xenakis and Olivier Messiaen. *Arts-sciences, alloys: the thesis defense of Iannis Xenakis before Olivier Messiaen, Michel Ragon, Olivier Revault d'Allonnes, Michel Serres, and Bernard Teyssèdre*. Pendragon Press, 1985.

James McCartney. Continued Evolution of the SuperCollider Real Time Synthesis Environment. In *Proceedings of the International Computer Music Conference*, 1998.

James Tenney and Larry Polansky. Temporal Gestalt Perception in Music. *Journal of Music Theory*, 24(2):204–241, 1980.

Jason Freeman. Extreme Sight-Reading, Mediated Expression, and Audience Participation: Real-Time Music Notation in Live Performance. *Computer Music Journal*, 32(3):25–41, 2008.

John Chowning. The Synthesis of Complex Audio Spectra by Means of Frequency Modulation. *Journal of the Audio Engineering Society*, 21(3), 1973.

John Dunn and Mary Anne Clark. "Life Music": The Sonification of Proteins. *Leonardo*, 32(1):25–32, 1999.

Jon Dattorro. Effect Design, part 1: Reverberator and other filters. *Journal of the Audio Engineering Society*, 45(9):660–684, 1997.

Jorge Solis, Keisuke Chida, Koichi Taniguchi, and Shinichiro Michael Hashimoto. The Waseda Flutist Robot WF-4RII in Comparison with a Professional Flutist. *Computer Music Journal*, 30(4):12–27, 2006.

Jos. Smits van Waesberghe. The Musical Notation of Guido of Arezzo. *Musica Disciplina*, 5:15–53, 1951.

Joseph Horowitz. *Understanding Toscanini: A Social History of American Concert Life*. University of California Press, 1987.

F. Joseph Smith. *The Experiencing of Musical Sound: Prelude to a Phenomenology of Music*. Gordon and Breach, New York, 1979.

Julio D'Escrivan. Reflections on the Poetics of Time in Electroacoustic Music. *Contemporary Music Review*, 3(1):197–201, 1989.

Gottfried Michael Koenig. Working with 'Project 1' my experiences with computer composition. *Interface*, 20(3-4):175–180, 1991.

Kyle Gann. *The Music of Conlon Nancarrow*. Cambridge University Press, 2006.

Laura Maes, Godfried-Willem Raes, and Troy Rogers. The Man and Machine Robot Orchestra at Logos. *Computer Music Journal*, 35(4):28–48, 2011.

Lebar Bajec, N. Zimic, and M. Mraz. The computational beauty of flocking: boids revisited. *Mathematical and Computer Modelling of Dynamical Systems: Methods, Tools and Applications in Engineering and Related Sciences*, 13(4): 331–347, 2007.

George E. Lewis. Interacting with latter-day musical automata. *Contemporary Music Review*, 18(3):99–112, 1999.

David G. Malham. Toward Reality Equivalence in Spatial Sound Diffusion. *Computer Music Journal*, 25(4):31–38, 2001.

David G. Malham and Anthony Myatt. Sound Spatialization using Ambisonic Techniques. *Computer Music Journal*, 19(4):58–70, 1995.

Peter Manning. Computers and Music Composition. *Proceedings of the Royal Musical Association*, 107(1):109–131, 1980.

Margaret A. Boden and Ernest A. Edmonds. What is Generative Art? *Digital Creativity*, 20(1-2):21–46, June 2009.

Margaret Vardell Sandresky. The Golden Section in Three Byzantine Motets of Dufay. *Journal of Music Theory*, 25(2):291–306, 1981.

Mark Katz. *Capturing Sound: How Technology Has Changed Music*. University of California Press, 2004.

Marvin Trachtenberg. Artitechture and Music Reunited: A New Reading of Dufay's Nuper Rosarum Flores and the Cathedral of Florence. *Renaissance Quarterly*, 54(3):740–775, 2001.

James McCartney. Rethinking the Computer Music language: Super Collider. *Computer Music Journal*, 26(4):61–68, 2002.

Matteo Meneghini. An Analysis of the Compositional Techniques in John Chowning's Stria. *Computer Music Journal*, 31(3):26–37, 2007.

Michael Edwards. Computational Thinking in Music. *Communications of the Association for Computing Machinery*, 54(7):58–67, 2011.

David P. Miller. Indeterminacy and Performance Practice in Cage's Variations. *American Music*, 27(1):60–86, 2009.

Miller S. Puckette, Apel Theodore, and David D. Zicarelli. Real-time audio analysis tools for Pd and MSP. In *Proceedings of the International Computer Music Conference*, 1998.

Eduardo Reck Miranda. Granular Synthesis of Sounds by Means of a Cellular Automaton. *Leonardo*, 28(4):297–300, 1995.

Eduardo Reck Miranda, Larry Bull, Francois Gueguen, and Ivan S. Uroukov. Computer Music Meets Unconventional Computing: Towards Sound Synthesis with In Vitro Neuronal Networks. *Computer Music Journal*, 33(1):9–18, 2009.

Eduardo Reck Miranda, Alexis Kirke, and Qijun Zhang. Artificial Evolution of Expressive Performance of Music: An Imitative Multi-Agent Systems Approach. *Computer Music Journal*, 34(1):80–96, 2010.

Mitchell Wand. What is Lisp. *The American Mathematical Monthly*, 91(1), 1984.

James A. Moorer. About This Reverberation Business. *Computer Music Journal*, 3(2):13–28, 1979.

Morag Josephine Grant. Experimental Music Semiotics. *International Review of the Aesthetics and Sociology of Music*, 34(2):173–191, 2003.

Keith Muscutt and David Cope. Composing with Algorithms: An Interview with David Cope. *Computer Music Journal*, 31(3):10–22, 2007.

Nick Collins. Infinite Length Pieces: A User's Guide. In *Proceedings of MAXIS, Sheffield Hallam University*, 2002.

Nick Collins and Andrew R. Brown. Generative Music Editorial. *Contemporary Music Review*, 28(1):1–4, 2009.

Nicolas Slonimsky. *Thesaurus of Scales and Melodic Patterns.* Amsco, 1986.

Noriko Manabe. New Technologies, Industrial Structure, and the Consumption of Music in Japan. *Asian Music*, 39(1):81–107, 2008.

Oded Ben Tal and Jonathan Berger. Creative Aspects of Sonification. *Leonardo*, 37(3):229–232, 2004.

Olli Niemitalo. Polynomial Interpolators for High-Quality Resampling of Oversampled Audio, 2001.

Heinz-Otto Peitgen, Hartmut Juergens, and Dietmar Saupe. *Chaos and Fractals: New Frontiers of Science.* Springer, New York, 2004.

Peter Bowcott. Cellular Automation as a Means of High Level Compositional Control of Granular Synthesis. In *Proceedings of the International Computer Music Conference*, 1989.

Peter Hoffmann. The New GENDYN Program. *Computer Music Journal*, 24(2): 31–38, 2000.

Peter Traub. Sounding the net: Recent sonic works for the internet and computer networks. *Contemporary Music Review*, 24(6):459–481, 2005.

Philip Carlsen. *The Player-Piano Music of Conlon Nancarrow: An Analysis of Selected Studies.* Institute for Studies in American Music, Conservatory of Music, Brooklyn College of the City University of New York, 1988.

Miller Puckette. Max at Seventeen. *Computer Music Journal*, 26(4):31–43, 2002.

Reebee Garofalo. From Music Publishing to MP3: Music and Industry in the Twentieth Century. *American Music*, 17(3):318–354, 1999.

Richard L. Crocker. Pythagorean Mathematics and Music. *The Journal of Aesthetics and Art Criticism*, 22(2):189–198, 1963.

Richard L. Crocker. Pythagorean Mathematics and Music. *The Journal of Aesthetics and Art Criticism*, 22(3):325–335, 1964.

Jean-Claude Risset. Composing in real-time? *Contemporary Music Review*, 18 (3):31–39, 1999.

Curtis Roads. Automated Granular Synthesis of Sound. *Computer Music Journal*, 2(2):61–62, 1978.

Curtis Roads. The Tsukuba Musical Robot. *Computer Music Journal*, 10(2): 39–43, 1986.

Curtis Roads. *Microsound*. The MIT Press, Cambridge, Massachusetts, 2001.

Robert Erickson. *Sound Structure in Music*. University of California Press, 1975.

Robin Maconie. Stockhausen's Mikrophonie I: Perception in Action. *Perspectives of New Music*, 10(2):92–101, 1972.

Robert Rowe. Machine Listening and Composing with Cypher. *Computer Music Journal*, 16(1):43–63, 1992.

Robert Rowe. *Machine Musicianship*. The MIT Press, Cambridge, Massachusetts; London, England, 2001.

Gunther Schuller and Edgard Varèse. Conversation with Varèse. *Perspectives of New Music*, 3(2):32–37, 1965.

Agostino Di Scipio. Compositional Models in Xenakis' Electroacoustic Music. *Perspectives of New Music*, 36(2):201–243, 1998.

Marie-Hélène Serra. Stochastic Composition and Stochastic Timbre: GENDY3 by Iannis Xenakis. *Perspectives of New Music*, 31(1):236–257, 1993.

Simon Atkinson. Interpretation and musical signification in acousmatic listening. *Organised Sound*, 12(2):113–122, 2007.

Steve Jones. Music and the Internet. *Popular Music*, 19(2):217–230, 2000.

Marco Stroppa. Live electronics or... live music? Towards a critique of interaction. *Contemporary Music Review*, 18(3):41–77, 1999.

Bob L. Sturm. Composing for an ensemble of atoms: the metamorphosis of scientific experiment into music. *Organised Sound*, 6(2):131–145, 2001.

Suzanne K. Roffler and Robert A. Butler. Localization of Tonal Stimuli in the Vertical Plane. *Journal of the Acoustical Society of America*, 43(6):1260–1266, 1968.

Tak Wing Chan and John H. Goldthorpe. Social Stratification and Cultural Consumption: Music in England. *European Sociological Review*, 23(1):1–19, 2007.

Theodor W. Adorno. *The Culture Industry*. Routledge, 1991.

Thor Magnusson. Algorithms as Scores: Coding Live Music. *Leonardo Music Journal*, 21:19–23, 2011.

Timothy Warner. Sign O' the times: The Sampler. *Contemporary Music Review*, 15(3-4):49–55, 1996.

O. S. Trimble. Localization of Sound in the Anterior Posterior and Vertical Dimensions of Auditory Space. *British Journal of Psychology*, 24:320–334, 1934.

Maja Trochimczyk. From Circles to Nets: On the Signification of Spatial Sound Imagery in New Music. *Computer Music Journal*, 25(4):39–56, 2001.

Barry Truax. Real-Time Granular Synthesis with a Digital Signal Processor. *Computer Music Journal*, 12(2):14–26, 1988.

Barry Truax. Composing with Real-Time Granular Sound. *Computer Music Journal*, 20(2):120–134, 1990.

Volker Grimm, Elroy Revilla, Uta Berger, Florian Jeltsch, Wolf M. Mooij, Steven F. Railsback, Hans-Hermann Thulke, Jacob Weiner, Thorsten Wiegand, and

Donald L. DeAngelis. Pattern-Oriented Modelling of Agent-Based Complex Systems: Lessons from Ecology. *Science*, 310(5750):987–991, 2005.

Walter Benjamin. *The Work of Art in the Age of Mechanical Reproduction*. Penguin Books (London), 2008.

Gil Weinberg and Scott Driscoll. Toward Robotic Musicianship. *Computer Music Journal*, 30(4):28–45, 2006.

William A. Burnson and Hans G. Kaper. Automatic Notation of Computer-Generated Scores for Instruments, Voices and Electro-Acoustic Sounds. In *Proceedings of the International Computer Music Conference*, 2010.

Iannis Xenakis. The Origins of Stochastic Music. *Tempo*, (78):9–12, 1966.

Iannis Xenakis. Towards a Metamusic. *Tempo*, (93):2–19, 1970.

Iannis Xenakis. *Formalised Music*. Harmonologia. Pendragon Press, Hillsdale, NY, 1992.

Iannis Xenakis and John Rahn. Sieves. *Perspectives of New Music*, 28(1):58–78, 1990.

Iannis Xenakis, Roberta Brown, and John Rahn. Xenakis on Xenakis. *Perspectives of New Music*, 25(2):16–63, 1987.

Laura Zattra. The Assembling of Stria by John Chowning: A Philogical Investigation. *Computer Music Journal*, 31(3):38–64, 2007.

7 Appendices

7.1 CD Contents

7.1.1 CD A

1. Warblers (2009)

2. Hermit (2011)

3. Prime Pattern 33 (2009)

4. SnSu (2011)

7.1.2 CD B

1. pp (2011)

2. For Putten (2012)

3. Stratus (2012)

4. Traurig? (2010)

michael murphy

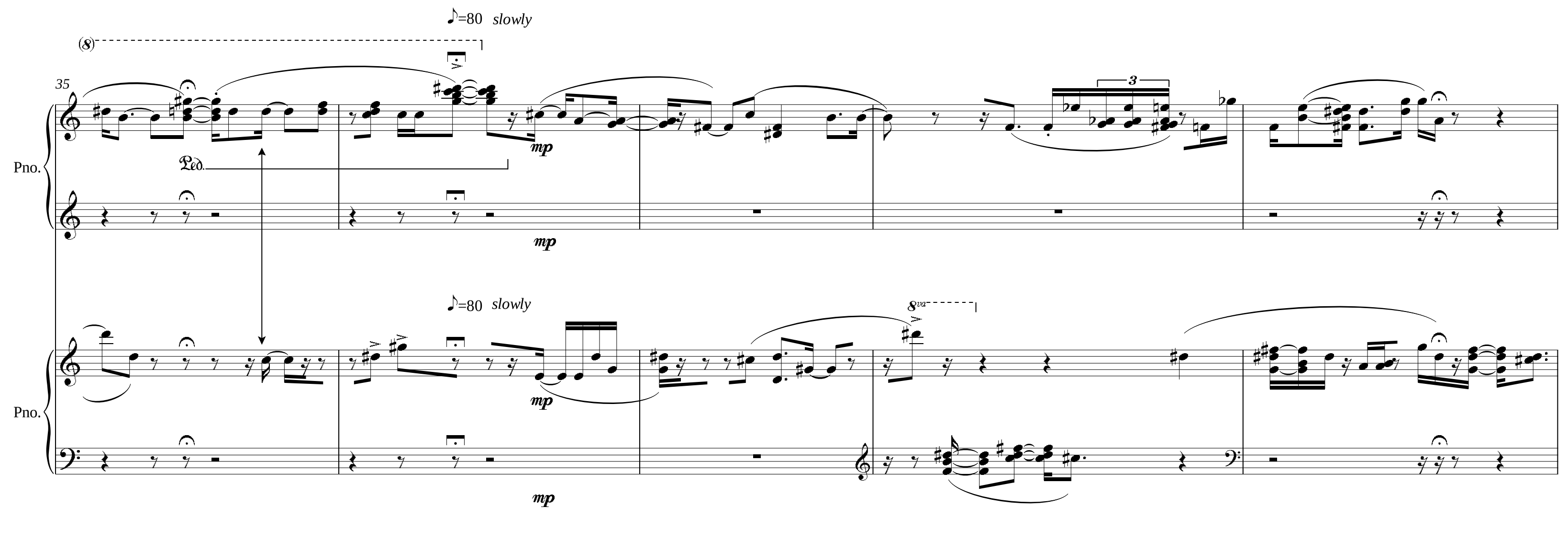
warblers(bonelli's, grasshopper)

serene
Piano
una corda
A
= 90 vocal, bright, conversational
Pno.


13
Pno.
Pno.
17
Pno.
Pno.


8va
22
Pno.
f
mf
15ma
Pno.
26
(15)
B
tumbling
ff
mp


30
Pno.
Pno.
mf
mf
mf
mf
pp
pp
pp


35
♪=80 slowly
♪=80 slowly
Pno.
Pno.
mp
mp
mp
mp


40
♩=70
mf
p
mp
Pno.
C
♩=90 energetic
8va
f
Pno.
45
(8)
ff
Pno.
Ped.
lurch
Pno.


48
Pno.
Pno.
52
Pno.
Pno.


56
Pno.
Pno.
mf
Ped.
15ma
3
mf
60
(15)
Pno.
Pno.
mp
Ped.
15ma

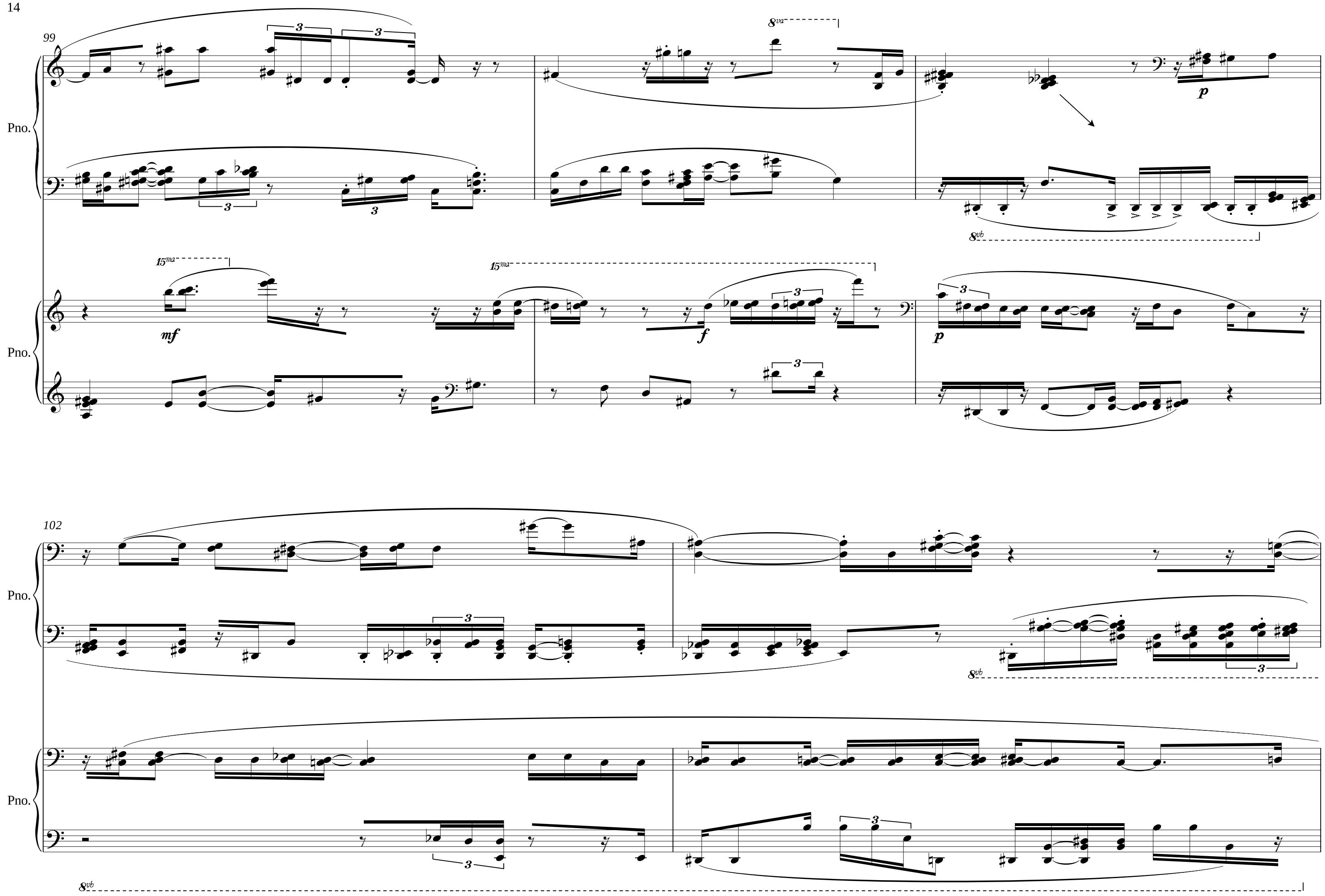

64
15ma
Pno.
cluster
Pno.
69
(15)
D ♩=70 fragmented
fff
mf
8vb
Ped.
fff
D ♩=70 fragmented
fff
mf
mf
Ped.


73
Pno.
Pno.
mp
mp
mf
mp
mp
mf
mp
8va
8vb
77
E
dense, blunt
Pno.
p
p
E
dense, blunt
Pno.
p
(8)
Ped.
p
3


phrases rise out of dense backdrop
81
Pno.
Pno.
85
Pno.
Pno.


88
Pno.
15ma
f
p
Pno.
Ped.
92
Pno.
3
Pno.
8va
p


95
Pno.
mf
8va
(8)
Pno.
97
Pno.
8va
15ma
f
Pno.


99
Pno.
8va
p
8vb
15ma
mf
f
102
3


104
Pno.
(8)
Pno.
8vb
106
♩=90
Ped.
fff
F
sparse
♩=40
8va
mp
15ma


110
Pno.
Pno.
116
G
mechanical
Pno.
G
mechanical
Pno.
cluster

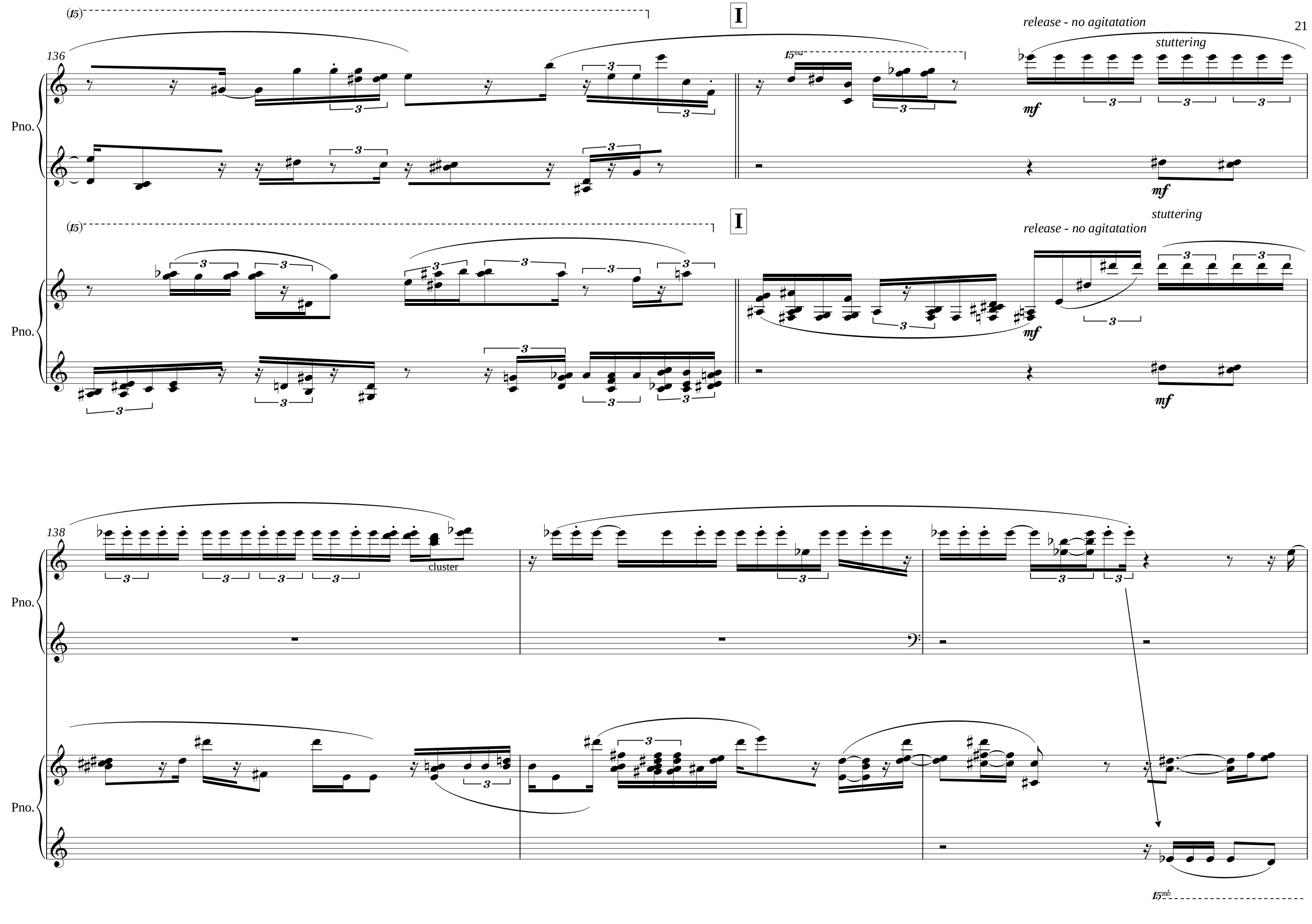

120
Pno.
mf
mf
8va
Pno.
mf
mf
8va
H
122
♩ = 90 all notes flowing into one, phrases emerge from dense background
Pno.
p
p
H
♩ = 90 all notes flowing into one, phrases emerge from dense background
Pno.
p
p


124
Pno.
mp
Pno.
126
15ma
increasing agitation/excitement till i
Pno.
p
mf
increasing agitation/excitement till i
Pno.
mf
p

128
Pno.
Pno.
130
Pno.
Pno.

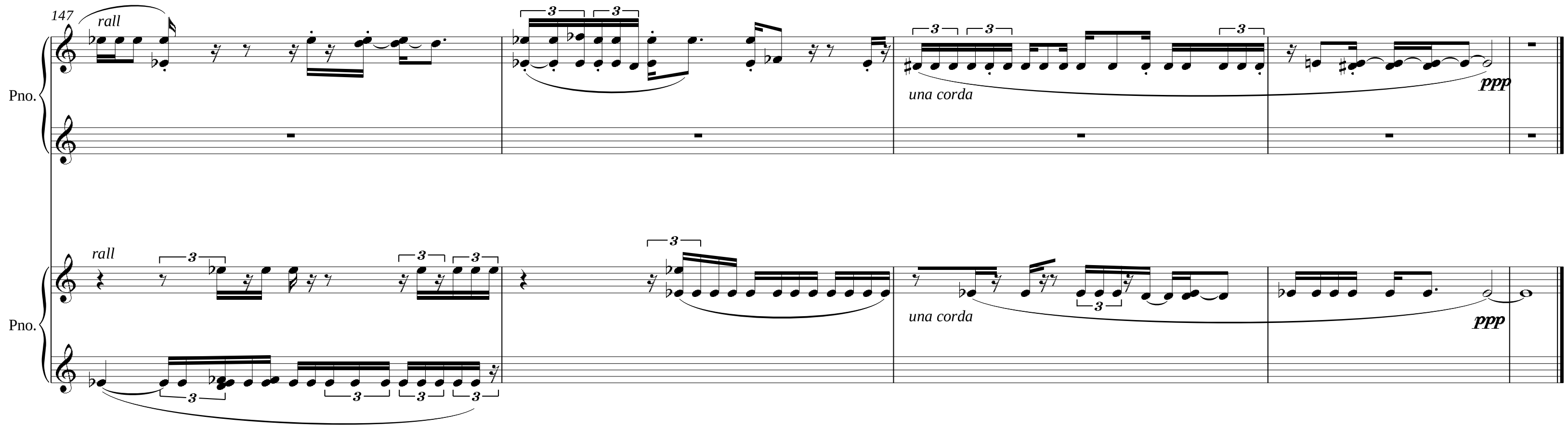

132
Pno.
Pno.
ff
ff
ff
ff
8va


134
(8)
15ma
Pno.
Pno.
fff
fff
fff
fff
15ma



136
(15)
I
15ma
release - no agitatation
stuttering
mf
Pno.
138
cluster
15mb


141
Pno.
Pno.
(15)
144
Pno.
Pno.


147
rall
Pno.
una corda
ppp
rall
Pno.
una corda
ppp

Sneaky

Michael Murphy


10
Pno.
Pno.
12
Pno.
Pno.
14
Pno.
Pno.
8va


17
Pno.
Pno.
mp
pp
mp
pp
p
p
18
8va
Pno.
Pno.
pp
p
p
pp
mf
p
21
Pno.
Pno.
mp
p
pp
mf
p
pp
pp
p
mp
mf
p
pp


26
Pno.
Pno.
30
Pno.
Pno.
33
Pno.
Pno.


36
Pno.
Pno.
40
Pno.
8va
Pno.
44
Pno.
Pno.


48
Pno.
pp
mp
p
mf
Pno.
51

Tinnitus

ca. 1"

M. Murphy 2010


Pno.
Pno.
Pno.
Pno.
Pno.
Pno.


8
Pno.
mf f p
mf pp mf p f
Pno.
f pp
f mf f
f
mf
10
Pno.
mf p f p pp ppp mf pp
pp
Pno.
p mf p
ppp
11
Pno.
mf ppp f p f pp f
Pno.
mf f mf
p

12
Pno.
pp ppp f pp
Pno.
f mf p mf
f
13
Pno.
f ppp mf p f mf f
p
Pno.
pp f pp f
p
14
Pno.
pp f p mf
p
Pno.
mf pp mf f mf f
p f p


15
Pno.
Pno.
16
Pno.
Pno.
17
Pno.
Pno.


18
Pno.
Pno.
mf
f
p
f
mf
f
pp
fff
f
p
pp
mf
f
pp
f
20
espress.
Pno.
Pno.
ppp
pp
mf
mf
pp
p
ppp
pp
mf
22
Pno.
Pno.
p
pp
ppp
mf
mf
ppp


23
Pno.
Pno.
24
Pno.
Pno.
25
Pno.
Pno.


26
Pno.
Pno.
28
Pno.
Pno.
29
Pno.
Pno.

Crescendi/Glissandi

Michael Murphy

(8)
4
Pno.
mf
mp
p
pp
Pno.
5
6


7
8va
Pno.
pp
mf
pp
mp
p
mp
pp
8
Pno.
mf
mp
pp
8vb
f
pp
mf
p
mf
9
Pno.
mf
mp
pp
p
pp
mf
mf
mp
pp
pp
p


10
Pno.
Pno.
11
Pno.
Pno.
12
Pno.
Pno.


13
Pno.
pp
mf
8vb
mf
Pno.
p
mp
pp
pp
mf
14
Pno.
pp
mf
p
mp
mf
Pno.
mf
mp
mf
15
Pno.
mp
pp
Pno.
pp
f


16
Pno.
p
mf
mf
pp
mf
pp
Pno.
p
ppp
mf
mp
mp
decresc.
17
Pno.
p
ppp
p
pp
Pno.
pp
18
Pno.
mf
pp
Pno.
pp
mf


19
very quietly
Pno.
very quietly
Pno.
20
Pno.
Pno.
21
Pno.
8vb
Pno.


22
Pno.
Pno.
23
Pno.
Pno.
24
Pno.
Pno.
fff
ppp
pp
mf
mp
ff
p


25
Pno.
ppp
mf
p
f
p
f
Pno.
mp
mf
mf
f
mp
pp
26
Pno.
pp ff
pp
fff
mp
pp
mp
fff
Pno.
mf
pp
mp
ppp
mf
pp
mp
8vb
ff
pp
27
Pno.
p
mf
pp
Pno.
mp
fff
ppp
pp


28
Pno.
Pno.
29
Pno.
Pno.
30
Pno.
Pno.


31
Pno.
Pno.
32
Pno.
Pno.
33
Pno.
Pno.


35
Pno.
Pno.
36
Pno.
Pno.
37
Pno.
Pno.


Pno.
Pno.
Pno.
Pno.
Pno.
Pno.


42
Pno.
Pno.
mp
mp
p
mp
mp
ppp
p
mp
mp pp mp
p mp
43
Pno.
Pno.
mf
ppp
mf
pp
pp
mp
mf
ppp mp
pp
ppp mp
44
Pno.
Pno.
p
mf
mp
mf
mp
mf
ppp
pp

Contrast Undecided

M. Murphy 2010


6
Pno.
Pno.
7
Pno.
Pno.
8
Pno.
Pno.


10
Pno.
Pno.
13 ♩=80
Pno.
♩=80
Pno.
15
Pno.
Pno.


17
Pno.
Pno.
19 ♩=70
Pno.
♩=70
Pno.
21
Pno.
Pno.


Pno.
Pno.
Pno.
Pno.
Pno.
Pno.
Ped.
Ped.


Pno.
Pno.
31
35
38
Ped.
Ped.


41
Pno.
Pno.
43
Rall.
Pno.
Rall.
Pno.
46
Pno.
Pno.


51
Pno.
♩=60
Pno.
♩=60
54
Pno.
Pno.
58
Pno.
♩=80
8va
Pno.
♩=80


63
Pno.
Pno.
68
Pno.
Pno.
72
Pno.
Pno.
8va
pp mp ff ppp


77
Pno.
Pno.
82
Pno.
Pno.
84
Pno.
Pno.

Double Dip

Michael Murphy


Pno.
Pno.
15ma
8va
mf ff f mp ppp pp p


(8)
10
Pno.
Pno.
mf pp mp mf ppp mf
mf
mf
ff mp mf
11
Pno.
Pno.
f mf ff pp
f ppp
pp ff mp
ppp mf ff mp mf mp
12
Pno.
Pno.
pp ff
pp
8va
ff ppp mp
ff mf pp mf pp mp ff


13
Pno.
Pno.
Ped.
14
8va
Pno.
Pno.
15
(8)
Pno.
Pno.
Ped.


(8)
17
Pno.
ff
mf
Pno.
ff
mf
(8)
18
Pno.
pp
ff
ff
Pno.
ppp
ff
ff
pp
19
Pno.
p
mf
Pno.
mp
ff
mf
ff

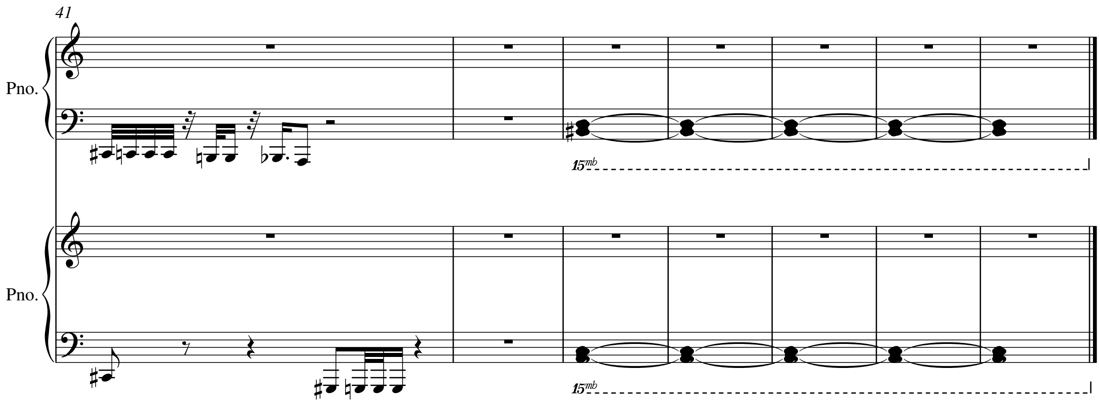

21
Pno.
Pno.
15ma
ff
ffff
fff
f
ff
p
22
(15)
Pno.
Pno.
f
ff
mf
pp
24
Pno.
Pno.
mf
ff
pp


26
15ma
8va
Pno.
mf
mp
ff
pp
Ped.
28
fff
29
ffff


30
Pno.
Pno.
15ma
8va
ppp
mp
mf
ff
pp
31
(15)
Ped.
32
fff


33
15ma
Pno.
Pno.
34
(15)
Pno.
Pno.
35
Pno.
Pno.


36
15ma
Pno.
mf
ff
8va
Pno.
mp
ff
f
39
Pno.
mf
ff
Pno.
pp
mf
ff
40
Pno.
fffff
Pno.
ffff
fffff


41
Pno.
Pno.
15mb
15mb

33

ca. 3min

Michael Murphy


21
Pno.
p
mf
Pno.
p
mf
3
f
mf
25
Pno.
rall.
♩=180
mf
mp
Pno.
rall.
♩=180
mp
29
Pno.
8va
mf
Pno.
mf
mf


33
Pno.
Pno.
mp
mp
mf
mp
mp
37
Pno.
Pno.
3
Ped.
Ped.
41
Pno.
Pno.
47
8va
Pno.
Pno.
p
p
p
p


52
Pno.
Pno.
56
Pno.
Pno.
60
ca. 4-5"
senza
Pno.
l.v.
Ped.
Pno.
senza
Ped.
67
Pno.
Pno.


72
Pno.
Pno.
76
Pno.
Ped.
Pno.
79
Pno.
Pno.


82
Pno.
Pno.
8va
85
Pno.
Pno.
8va
mp
mf
Ped.
90
accel.
Pno.
Pno.
mp
mf
Ped.
3
95
Pno.
Pno.
mf


(8)
♩=250
100
Pno.
Pno.
104
Ped.
8va
f
109
114


118
Pno.
Pno.
121
Pno.
Pno.
8va
125
Pno.
ff
Pno.
8va
128
(8)
Pno.
f
Pno.


131
Pno.
Pno.
134
♩=180
mf
pp
Pno.
♩=180
mf
Pno.
138
mp
l.v.
Pno.
l.v.
mp
Pno.